

**A STABLE ALGORITHM FOR DIVERGENCE-FREE
AND CURL-FREE RADIAL BASIS FUNCTIONS IN THE
FLAT LIMIT**

by

Kathryn Primrose Drake

A thesis

submitted in partial fulfillment
of the requirements for the degree of
Master of Science in Mathematics
Boise State University

August 2017

BOISE STATE UNIVERSITY GRADUATE COLLEGE

DEFENSE COMMITTEE AND FINAL READING APPROVALS

of the thesis submitted by

Kathryn Primrose Drake

Thesis Title: A Stable Algorithm for Divergence-Free and Curl-Free Radial Basis Functions in the Flat Limit

Date of Final Oral Examination: 02 June 2017

The following individuals read and discussed the thesis submitted by student Kathryn Primrose Drake, and they evaluated the presentation and response to questions during the final oral examination. They found that the student passed the final oral examination.

Grady B. Wright, Ph.D.

Chair, Supervisory Committee

Jodi Mead, Ph.D.

Member, Supervisory Committee

Donna Calhoun, Ph.D.

Member, Supervisory Committee

The final reading approval of the thesis was granted by Grady B. Wright, Ph.D., Chair of the Supervisory Committee. The thesis was approved by the Graduate College.

dedicated to Bodie

ACKNOWLEDGMENTS

I first express my gratitude to my advisor, Dr. Grady Wright. His constant guidance, patience, and enthusiasm helped me to become a better mathematician and researcher. Next I thank the other members of my committee, Dr. Jodi Mead and Dr. Donna Calhoun. Their instruction and accomplishments inspired me to challenge myself and persist. I am also grateful to the Boise State University Mathematics Department and Graduate College for the funding that supported this work.

I have been immeasurably fortunate to have family members that love and support me. Special thanks goes to my mother, Jennifer. Her love has been the foundation upon which I have built my character. My friends have provided endless light and laughter throughout my life, which has been especially meaningful during my time in this program. Thank you to Kayla and Kara, whose friendships formed my childhood and continue to encourage me every day. I am also sincerely grateful to my fellow math graduate students. Our camaraderie allowed us to form a bond that I will always cherish.

Finally, I thank my husband, Bodie. You made my dreams your own and then you helped make them a reality. You fill every day with joy and every journey with adventure. I cannot imagine walking this road with a better companion and friend.

ABSTRACT

Radial basis functions (RBFs) were originally developed in the 1970s for interpolating scattered topographic data. Since then they have become increasingly popular for other applications involving the approximation of scattered, scalar-valued data in two and higher dimensions, especially data collected on the surface of a sphere. In the late 2000s, matrix-valued RBFs were introduced for approximating divergence-free and curl-free vector fields on the surface of a sphere from scattered samples, which arise naturally in atmospheric and oceanic sciences. The intriguing property of these RBFs is that the resulting vector-valued approximations analytically preserve the divergence-free or curl-free properties of the field.

The most commonly used RBFs feature a shape parameter that controls how peaked or flat the basis functions are, with the choice of this parameter greatly affecting the accuracy of the RBF approximation to the underlying data. Flatter basis functions, which correspond to small shape parameters, generally result in more accurate approximations when the sampled data comes from a smooth function or vector-field. However, the direct method for computing the resulting RBF approximation becomes horribly ill-conditioned as the basis functions are made flatter and flatter. For scalar-valued RBF approximation, this was a fundamental issue until the mid-2000s when researchers started to develop stable algorithms for “flat” RBFs. One of the most successful of these is the RBF-QR algorithm, which completely bypasses the ill-conditioning associated with flat scalar-valued RBFs on the sphere using a clever change of basis. In this thesis, we extend the RBF-QR algorithm to

flat matrix-valued RBFs for approximating both divergence-free and curl-free vector fields on the sphere. We give numerical results illustrating the effectiveness of this new algorithm and also show that in the limit where the matrix-valued RBFs become entirely flat, the resulting approximations converge to vector spherical harmonic approximants. This is the first algorithm that allows for stable computations of divergence-free and curl-free matrix-valued RBFs in the flat limit.

TABLE OF CONTENTS

ABSTRACT	vi
LIST OF TABLES	x
LIST OF FIGURES	xi
1 Background	1
1.1 Introduction	1
1.2 Radial Basis Function (RBF) Interpolation	3
1.2.1 RBF Interpolation of Scalar-valued Functions	4
1.2.2 RBF Interpolation of Vector Fields	9
1.3 RBF Interpolation of Surface Divergence-Free and Curl-Free fields on the Sphere	14
1.3.1 Surface Differential Operators for Vector Fields in \mathbb{R}^3	14
1.3.2 Vector RBF Interpolation on the Sphere	15
1.4 Spherical Harmonics	22
1.4.1 Scalar Spherical Harmonics	23
1.4.2 Vector Spherical Harmonics	24
1.5 Overview of the Thesis	26
2 The RBF-QR Algorithm for Stable Scalar-Valued RBF Interpolation on the Sphere	28

2.1	Scalar-Valued RBF Interpolation in the Flat Limit	28
2.2	Scalar RBF-QR Algorithm	31
2.2.1	Spherical Harmonic Expansion of RBF Kernels	31
2.2.2	Matrix Representation and QR Factorization	33
2.2.3	Numerical Results	38
2.2.4	The Size of n	39
3	Vector RBF-QR Algorithm	41
3.1	Vector-Valued RBF Interpolation in the Flat Limit	41
3.2	Vector RBF-QR Algorithm for Surface Divergence-Free RBFs	42
3.2.1	Vector Spherical Harmonic Expansion	42
3.2.2	Matrix Representation and QR Factorization	43
3.3	Vector RBF-QR Algorithm for Surface Curl-Free RBFs	49
3.4	Vector RBF-QR Algorithm for the Helmholtz-Hodge Decomposition of Surface Vector Fields	52
4	Numerical Results from the Vector RBF-QR Algorithm	53
4.1	Surface Divergence-Free Vector Fields	53
4.2	Surface Curl-Free Vector Fields	59
4.3	Conclusions	62
5	Conclusions	63
	REFERENCES	65
A	Proof of Lemma	68

LIST OF TABLES

1.1	Commonly used radial kernels, where the first three are positive definite, $r = \ \mathbf{x} - \mathbf{y}\ $, and ε is the shape parameter.	7
2.1	SPH expansion coefficients for various radial kernels on the sphere. In the formula for the IQ kernel, ${}_2F_1(\dots)$ denotes the hypergeometric function, and in the formula for the GA kernel, $I_{\mu+1/2}$ denotes the Bessel function of the second kind. Note that the apparent singularity of the $c_{\mu,\varepsilon}$ for the GA kernel is a removable one due to the identity $\frac{I_{\mu+1/2}(2\varepsilon^2)}{\varepsilon^{2\mu+1}} = \frac{1}{\Gamma(\mu+1)\sqrt{\pi}} \int_{-1}^1 e^{2\varepsilon^2 t} (1-t^2)^\mu dt$	32

LIST OF FIGURES

1.1	The process of using RBFs to interpolate a set of scattered data in $2D$. (a) a target function f sampled at some set of distinct nodes, (b) a set of radial basis functions interpolating the data (c) a reconstructed surface resulting from the interpolation	7
1.2	(a) The Gaussian ($\varepsilon = 2$), (b) inverse quadric ($\varepsilon = 3.5$), (c) inverse multiquadric ($\varepsilon = 6$), and (d) multiquadric radial kernels ($\varepsilon = 2$).	8
1.3	The columns of a divergence-free kernel: (a) $\Phi_{div}(\mathbf{x}, 0)[1 \ 0]^T$ (b) $\Phi_{div}(\mathbf{x}, 0)[0 \ 1]^T$ based on the Gaussian radial kernel.	11
1.4	(a) The samples of a divergence-free vector field and (b) the interpolant of the field using the Gaussian kernel with $\varepsilon = 4.5$	12
1.5	The columns of a curl-free kernel: (a) $\Phi_{curl}(\mathbf{x}, 0)[1 \ 0]^T$ (b) $\Phi_{cur}(\mathbf{x}, 0)[0 \ 1]^T$ based on the Gaussian radial kernel.	13
1.6	(a) The samples of a curl-free vector field and (b) the interpolant of the field using the Gaussian kernel with $\varepsilon = 4.5$	13
1.7	The two components of the tangent vector basis at \mathbf{y}_j : (a) Zonal basis, $\Psi_{div}(\mathbf{x}, \mathbf{y}_j)\mathbf{e}_j$ (b) Meridional basis, $\Psi_{div}(\mathbf{x}, \mathbf{y}_j)\mathbf{d}_j$	18
1.8	(a) The scattered samples of a surface divergence-free vector field in blue and (b) the interpolant of the field using the surface divergence-free RBF interpolant in black.	19

1.9	The two components of the tangent vector basis at \mathbf{y}_j : (a) Zonal basis, $\Psi_{curl}(\mathbf{x}, \mathbf{y}_j)\mathbf{e}_j$ (b) Meridional basis, $\Psi_{curl}(\mathbf{x}, \mathbf{y}_j)\mathbf{d}_j$	20
1.10	(a) The scattered samples of a surface curl-free vector field in red and (b) the interpolant of the field using the surface curl-free RBF interpolant in black.	21
1.11	Pseudocolor plot of the scalar spherical harmonics basis functions of degrees $\mu = 0, 1, 2, 3, 4$ and orders $\nu = -\mu, \dots, \mu$. The colors range from blue to red, which correspond to negative and positive values, respectively.	24
2.1	The inverse multiquadric kernel for (a) $\varepsilon = 10$, (b) $\varepsilon = 5$, and (c) $\varepsilon = 1$	28
2.2	A problem illustrating ill-conditioning that enters the interpolation process in the RBF Direct method for an interpolation problem on the sphere consisting of (a) $n = 529$ quasi-uniformly distributed nodes and (b) the target function $f = \sin(xyz)$ on the sphere. (c) Condition number of the A_Y matrix in (1.6) vs ε . (d) Max norm error vs ε in the resulting RBF interpolant over the sphere computed with the RBF Direct approach. The IMQ kernel was used here.	30
2.3	Log-log plot of the max norm error vs. values of ε for the target function $f = \sin(xyz)$. Here $n = 529$, and the IMQ RBF kernel was used. Note that for larger values of ε , the RBF Direct and RBF-QR methods give equivalent results. Though not clearly visible in the figure, this equivalence is demonstrated where the black line lies on top of the dashed red line.	38

3.1	A problem illustrating the ill-conditioning that enters the interpolation process in the RBF Direct method for $n = 528$ quasi-uniformly distributed nodes and target function Ψ used in the second numerical test in Chapter 4. (a) Condition number of the $A_{\Psi_{div}}$ matrix from (1.23) vs. ε (b) Max norm error vs ε in the surface divergence-free RBF interpolant using the RBF Direct approach	42
4.1	Minimum energy node sets used in the numerical experiments: (a) 120 nodes and (b) 528 nodes	54
4.2	The surface divergence-free vector field to be interpolated for test 1. . . .	54
4.3	Numerical test 1: Log-log plot of the max-norm error in the approximation of the true field vs values of ε for both the RBF Direct method and the Vector RBF-QR method with the (a) MQ and (b) IMQ kernels. . . .	55
4.4	The surface divergence-free vector field to be interpolated in test 2. . . .	56
4.5	Numerical test 2: Log-log plot of the max-norm error in the approximation of the true field vs values of ε for both the RBF Direct method and the Vector RBF-QR method with the (a) MQ and (b) IMQ kernels. . . .	57
4.6	The surface divergence-free vector field to be interpolated for test 3. . . .	58
4.7	Numerical test 3: Log-log plot of the max-norm error in the approximation of the true field vs values of ε for both the RBF Direct method and the Vector RBF-QR method with the (a) MQ and (b) IMQ kernels. . . .	58
4.8	The surface curl-free vector field to be interpolated for test 1.	59

4.9	Curl-free numerical test 1: Log-log plot of the max-norm error in the approximation of the true field vs values of ε for both the RBF Direct method and the Vector RBF-QR method with the (a) MQ and (b) IMQ kernels.	60
4.10	The surface curl-free vector field to be interpolated for test 2.	61
4.11	Curl-free numerical test 2: Log-log plot of the max-norm error in the approximation of the true field vs values of ε for both the RBF Direct method and the Vector RBF-QR method with the (a) MQ and (b) IMQ kernels.	61

CHAPTER 1

BACKGROUND

1.1 Introduction

The interpolation of scattered data is a problem that emerges in multiple scientific disciplines and applications, such as meteorology, electronic imaging, computer graphics, medicine, and the Earth sciences [1, 9, 19, 25, 27]. Radial Basis Functions (RBFs) were first introduced in 1968 by R.L. Hardy to solve a $2D$ scattered data interpolation problem in cartography [15, 16]. Named for their use of linear combinations of shifts of radially symmetric functions to interpolate data and approximate surfaces, these scalar-valued RBFs have been used for various applications in fields ranging from geophysics to statistics [10, 24]. Part of the usefulness behind RBFs is their inclusion of a “shape parameter,” which controls the peakedness of the basis functions. Researchers have observed that this shape parameter directly impacts the accuracy of the target function approximation. Specifically, they noted that smaller values of the shape parameter result in better approximations to a point at which severe ill-conditioning enters into the system of equations for determining the interpolation coefficients. While there is a substantial amount of literature dedicated to finding the “optimal” shape parameter for scalar-valued RBFs [4, 26], these methods have been restricted by this ill-conditioning that is introduced into the system when the shape parameter approaches zero (named the “flat limit”) [5].

In 2007, Fornberg and Piret developed the RBF-QR algorithm, which bypassed the ill-conditioning of scalar RBF interpolation on the sphere in the flat limit [7]. They achieved this through a clever use of spherical harmonic expansions and the QR factorization to create a new set of basis functions that span the same space as the RBF basis, but are well-conditioned in the flat limit. While this method allowed for the full range of the shape parameter to be explored for scalar-valued RBFs on the sphere, it did not directly apply to vector-valued RBF interpolants, which are used to approximate vector fields.

Vector fields arise in many scientific applications, as they describe certain fundamental physical quantities. Specifically, there are two properties of vector fields that are useful when representing physical data: divergence (sources and sinks in a field) and curl (rotational movement of a field). RBF interpolation theory has been extended for use of interpolating both divergence-free and curl-free vector fields in \mathbb{R}^d [11, 22] and on the sphere [13, 23], but these vector-valued interpolants suffer the same dependence on the shape parameter.

This thesis introduces the first stable numerical method for calculating vector-valued RBF interpolants on the sphere in the flat limit. Our method, which we call the Vector RBF-QR algorithm, synthesizes existing derivations of vector-valued RBF interpolation on the sphere [13] and Fornberg and Piret’s stable algorithm for scalar-valued RBF interpolation in the flat limit [7]. Similar to the Scalar RBF-QR algorithm, the Vector RBF-QR algorithm utilizes vector spherical harmonic expansions and a QR factorization in order to create a better conditioned set of basis functions that span the same space as the standard vector basis used to construct the vector RBF interpolant.

The structure of this thesis is as follows. The remainder of this chapter gives an

overview of the background material needed for the algorithms presented in Chapters 2 and 3. This includes details of both scalar-valued and vector-valued RBF interpolation, as well as relevant information on scalar and vector spherical harmonics. Chapter 2 offers extensive detail of Fornberg and Piret’s RBF-QR algorithm for scalar-valued RBF interpolation, including numerical results. The main result of this thesis, the Vector RBF-QR algorithm, is presented in Chapter 3. We include numerical results from this algorithm in Chapter 4 and conclude with comments on future work in Chapter 5.

1.2 Radial Basis Function (RBF) Interpolation

As mentioned in the introduction, interpolating scattered data is a problem that arises in many disciplines, including engineering, hydrology, and geophysics. Several established techniques, like polynomial and trigonometric interpolation, have been used to solve this problem in one dimension. These methods typically use linear combinations from a fixed set of basis functions $\{\psi_j(x)\}_{j=1}^n$ to form a function $s(x)$ that will interpolate the data points $\{x_j\}_{j=1}^n$ at the values $\{f_j\}_{j=1}^n$. The resulting function is of the form

$$s(x) = \sum_{j=1}^n c_j \psi_j(x) \quad (1.1)$$

and must satisfy the interpolation conditions $s(x_j) = f_j$, for $j = 1, \dots, n$. These conditions lead to linear constraints on the expansion coefficients, c_j . These coefficients can be determined by solving the linear system of equations

$$\begin{bmatrix} A \end{bmatrix} \begin{bmatrix} \underline{c} \end{bmatrix} = \begin{bmatrix} \underline{f} \end{bmatrix}, \quad (1.2)$$

where the entries of A are given by $a_{j,k} = \psi_k(x_j)$ $j, k = 1, \dots, n$, \underline{c} is a vector containing the c_j 's, and \underline{f} is a vector containing the f_j 's. Many of these methods, including polynomial and trigonometric interpolation, work well in the one-dimensional case because this linear system is guaranteed to be non-singular whenever the given data are distinct [31]. However, this is not always the case for data in more than one dimension. The Mairhuber-Curtis theorem shows that for any set of basis functions that are independent of the data, there exist sets of distinct data points such that the linear system in (1.2) becomes singular [29]. In 1968, R.L. Hardy pioneered a solution to interpolating two-dimensional, scattered data in a way that bypasses the Mairhuber-Curtis theorem. He achieved this by developing a type of basis function that *depends on* the data rather than on a grid. This technique is known as the radial basis function (RBF) method.

1.2.1 RBF Interpolation of Scalar-valued Functions

The RBF method used today is a generalization of Hardy's multiquadric (MQ) method. The MQ method solved a common problem in cartography of finding a continuous function that accurately represents a surface given sparse measurements. Much of the motivation behind this investigation was twofold: to create contour maps of topographic surfaces and to use methods of calculus on the representative function to determine characteristics of the surface [15, 16]. Before working on a solution to this problem, Hardy first chose to investigate the one-dimensional version, i.e. finding a continuous function that accurately represents a curve given scattered data measurements. He first realized that a piecewise linear interpolating function provided a satisfactory representation of the curve, but discontinuities of the first derivative of the interpolating function prevented him from using calculus to analyze the curve.

After some trial and error, he finally chose to use the continuously differentiable basis function $\sqrt{a^2 + x^2}$, where $a \neq 0$ is an arbitrary constant that affects the peakedness of the quadric [15]. This choice leads to the interpolating function

$$s(x) = \sum_{j=1}^n c_j \sqrt{a^2 + (x - x_j)^2}, \quad (1.3)$$

where the c_j 's are determined as discussed in the previous section. Note that for $a = 0$, the basis function becomes piecewise linear.

Hardy found that there were many benefits to this MQ method beyond the fact that it accurately represented the desired curve. Specifically, he noticed that calculus techniques could be applied to the interpolating function (when $a \neq 0$) to gain meaningful information about the curve and more importantly, that this new approach could carry over to data in more than one dimension. This key property is what allowed Hardy to find a solution to his original cartography problem of approximating a surface with a mathematical function.

Hardy extended his one-dimensional interpolating technique first for data in $2D$. He achieved this by instead using a linear combination of quadric basis functions that were translated to be centered at each data point. For distinct points $\{(x_j, y_j)\}_{j=1}^n$ in \mathbb{R}^2 , the new interpolant is given by

$$s(x, y) = \sum_{j=1}^n c_j \sqrt{a^2 + (x - x_j)^2 + (y - y_j)^2}. \quad (1.4)$$

Hardy found that (1.4) performed well when used to approximate topographic surfaces from sparse data measurements. He named this technique the “multiquadric method” due to its most notable feature of “superpositioning quadric surfaces” [16]. While Hardy’s initial goal was achieved, he noted that his method could be easily extended for scattered data in many dimensions. His work, combined with the work of those

after him in the 1970s and 1980s, lead to the formal definition for scalar interpolation using RBFs:

Definition 1.2.1. (Scalar RBF Method) Given a distinct set of scattered nodes $Y = \{\mathbf{y}_j\}_{j=1}^n \subset \mathbb{R}^{d \geq 1}$ and some scalar-valued target function f sampled at Y , the scalar-valued RBF interpolant of $f|_Y$ is given by

$$s(\mathbf{x}) = \sum_{j=1}^n c_j \phi(\|\mathbf{x} - \mathbf{y}_j\|), \quad (1.5)$$

where $x \in \mathbb{R}^d$, $\|\cdot\|$ is the d-dimensional Euclidean norm, and $\phi(r)$ is some radial kernel (see Table 1.1 for examples). The expansion coefficients c_j can be determined by solving the symmetric linear system formed by enforcing the interpolation conditions $s(\mathbf{y}_j) = f_j$, $j = 1, \dots, n$:

$$\underbrace{\begin{bmatrix} \phi(\|\mathbf{y}_1 - \mathbf{y}_1\|) & \phi(\|\mathbf{y}_1 - \mathbf{y}_2\|) & \cdots & \phi(\|\mathbf{y}_1 - \mathbf{y}_n\|) \\ \phi(\|\mathbf{y}_2 - \mathbf{y}_1\|) & \phi(\|\mathbf{y}_2 - \mathbf{y}_2\|) & \cdots & \phi(\|\mathbf{y}_2 - \mathbf{y}_n\|) \\ \vdots & \vdots & \ddots & \vdots \\ \phi(\|\mathbf{y}_n - \mathbf{y}_1\|) & \phi(\|\mathbf{y}_n - \mathbf{y}_2\|) & \cdots & \phi(\|\mathbf{y}_n - \mathbf{y}_n\|) \end{bmatrix}}_{A_Y} \underbrace{\begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix}}_{\underline{c}} = \underbrace{\begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_n \end{bmatrix}}_{\underline{f}}. \quad (1.6)$$

We note here that determining the interpolation coefficients in this manner will be referred to as “RBF Direct” in this thesis. Geometrically, the RBF Direct method can be viewed as interpolating the data with a linear combination of translates of a single basis function, $\phi(r)$, that is radially symmetric about its center. This process can be seen graphically in Figure 1.1. Several options for these radial kernels have been developed since Hardy’s multiquadric kernel, and this thesis will use those with the following property.

Definition 1.2.2. (Positive Definite Kernel) Let $\Omega \subset \mathbb{R}^{d \geq 1}$. ϕ is a positive definite kernel on Ω if the matrix A_Y is positive definite for any distinct $Y = \{\mathbf{y}_j\}_{j=1}^n \subset \Omega$,

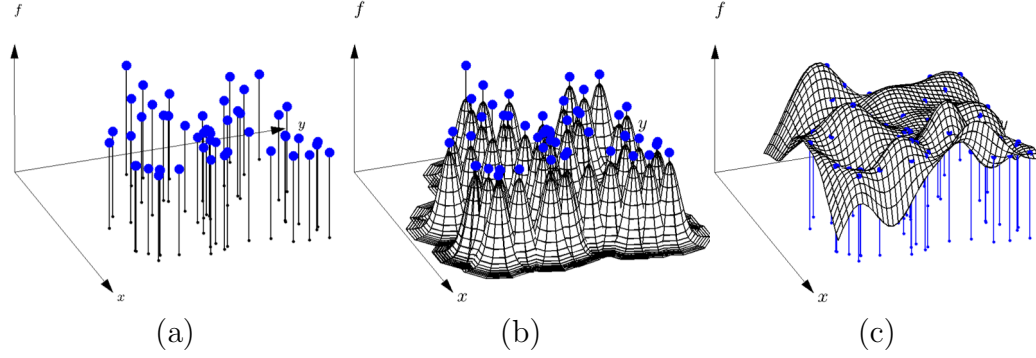


Figure 1.1: The process of using RBFs to interpolate a set of scattered data in $2D$. (a) a target function f sampled at some set of distinct nodes, (b) a set of radial basis functions interpolating the data (c) a reconstructed surface resulting from the interpolation

i.e.

$$\sum_{i=1}^n \sum_{j=1}^n b_i \phi(\mathbf{y}_i, \mathbf{y}_j) b_j > 0, \quad \text{provided } b_i \neq 0, \quad i = 1, \dots, n.$$

Table 1.1 lists some of the most commonly used, positive definite radial kernels, and Figure 1.2 shows plots of these kernels. Using these kernels guarantees that the A_Y matrix in (1.6) will be unconditionally nonsingular, i.e., that the RBF Direct method will be uniquely solvable [20]. Notice that the MQ kernel is precisely the one

Radial Kernel	$\phi(r)$
Gaussian (GA)	$e^{-(\varepsilon r)^2}$
Inverse quadratic (IQ)	$\frac{1}{1 + (\varepsilon r)^2}$
Inverse multiquadric (IMQ)	$\frac{1}{\sqrt{1 + (\varepsilon r)^2}}$
Multiquadric (MQ)	$\sqrt{1 + (\varepsilon r)^2}$

Table 1.1: Commonly used radial kernels, where the first three are positive definite, $r = \|\mathbf{x} - \mathbf{y}\|$, and ε is the shape parameter.

that Hardy developed with the transformation $a = \frac{1}{\varepsilon}$. Here, ε is a free parameter that controls the flatness or peakedness of the functions, giving it the name “shape

parameter.” The shape parameter plays a central role in this thesis and will be discussed in more detail in subsequent chapters.

Since its introduction by Hardy, the scalar-valued RBF interpolation method has been studied extensively for approximating scattered data in two and higher dimensions. RBFs have become increasingly popular and are now being used for

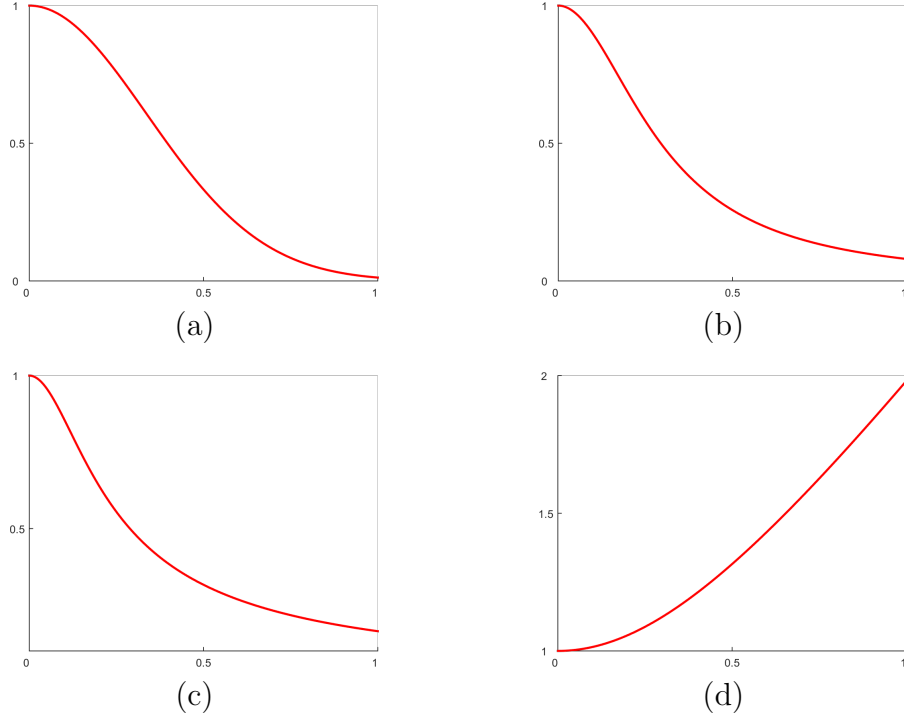


Figure 1.2: (a) The Gaussian ($\varepsilon = 2$), (b) inverse quadric ($\varepsilon = 3.5$), (c) inverse multiquadric ($\varepsilon = 6$), and (d) multiquadric radial kernels ($\varepsilon = 2$).

applications such as computer animation, medical imaging, and fluid dynamics. This section has given a brief overview of scalar-valued RBF interpolation, including their origins in cartography. In the next section, we will cover how to use the radial kernels in Table 1.1 for approximating vector-valued functions, i.e. vector fields, with certain inherent properties.

1.2.2 RBF Interpolation of Vector Fields

Vector fields arise naturally in many applications, as they describe certain fundamental physical quantities. Two specific properties of vector fields are particularly useful when representing physical data: divergence (sources and sinks in a field) and curl (rotational movement of a field). For example, divergence-free vector fields represent incompressible fluid flows and (static) magnetic fields, while curl-free vector fields represent gravity fields and (static) electric fields. Since these properties play a key role in many applications, interest was stirred in finding a method that used the established scalar-valued radial kernels to construct vector-valued approximations to vector fields with divergence-free or curl-free properties. We note here that many people consider a naïve approach for using RBFs to interpolate data collected from vector fields: using the scalar-valued RBF method to interpolate each component of the vector field individually. The disadvantage to this technique is that it does not allow certain properties of the vector field to be preserved. In order to properly utilize scalar-valued kernels to approximate vector fields while still preserving divergence-free and curl-free properties, researchers worked to extend the RBF theory for interpolating all components of a vector field together. Narcowich and Ward accomplished this in 1994 when they introduced matrix-valued kernels that can be used to produce divergence-free interpolants at scattered points [22]. In 2006, Fuselier did the same for curl-free interpolants [11].

The interpolation process for matrix-valued kernels is similar to that of the scalar case. We begin with divergence-free interpolation. Let ϕ be any scalar-valued radial kernel that is twice continuously differentiable, then we consider the matrix-valued, divergence-free kernel formed from ϕ as

$$\Phi_{div}(\mathbf{x}, \mathbf{y}) = -I\Delta\phi(\mathbf{x}, \mathbf{y}) + \nabla\nabla^T\phi(\mathbf{x}, \mathbf{y}), \quad (1.7)$$

where ∇ is the gradient operator in \mathbb{R}^d , $\nabla\nabla^T$ is the Hessian operator, and Δ is the Laplacian operator. In $2D$, for example,

$$-I\Delta = \begin{bmatrix} -\Delta & 0 \\ 0 & -\Delta \end{bmatrix} \quad \text{and} \quad \nabla\nabla^T = \begin{bmatrix} \partial_{xx} & \partial_{xy} \\ \partial_{yx} & \partial_{yy} \end{bmatrix}.$$

Narcowich and Ward showed that Φ_{div} is a $d \times d$ matrix-valued function with divergence-free columns [22]. In order to demonstrate why the columns of this kernel are divergence-free, we utilize the standard basis vectors $\mathbf{e}_j \in \mathbb{R}^d$. The j^{th} column of Φ_{div} is given by

$$\begin{aligned} \Phi_{div}(\mathbf{x}, \mathbf{y})\mathbf{e}_j &= [-I\Delta\phi(\mathbf{x}, \mathbf{y}) + \nabla\nabla^T\phi(\mathbf{x}, \mathbf{y})]\mathbf{e}_j \\ &= \nabla \times (\nabla \times (\phi(\mathbf{x}, \mathbf{y})\mathbf{e}_j)) = \nabla \times \mathbf{g}, \end{aligned} \quad (1.8)$$

where \mathbf{g} is a vector field. Thus we see that each column of Φ_{div} is the curl of a vector field, so they are divergence-free; see Figure 1.3 for an illustration of the columns of Φ_{div} in \mathbb{R}^2 . Additionally, Narcowich and Ward showed how to use Φ_{div} to produce a divergence-free vector-valued interpolant.

Given a distinct set of scattered nodes $Y = \{\mathbf{y}_j\}_{j=1}^n$ on \mathbb{R}^d and a target vector field $\mathbf{u} : \mathbb{R}^d \rightarrow \mathbb{R}^d$ sampled at Y , the divergence-free vector RBF interpolant of \mathbf{u} is given by

$$\mathbf{s}(\mathbf{x}) = \sum_{j=1}^n \Phi_{div}(\mathbf{x}, \mathbf{y}_j)\mathbf{c}_j, \quad \mathbf{c}_j \in \mathbb{R}^d. \quad (1.9)$$

Similar to the scalar-valued interpolation method, the interpolation coefficients \mathbf{c}_j are determined by solving the linear system formed by enforcing the interpolation conditions, which can be written as

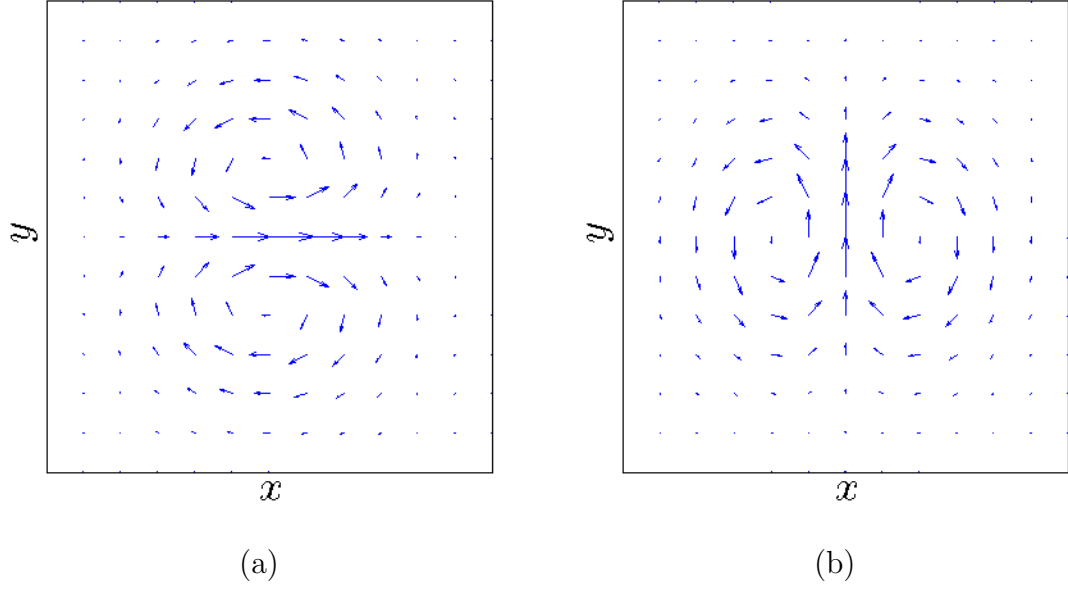


Figure 1.3: The columns of a divergence-free kernel: (a) $\Phi_{div}(\mathbf{x}, 0)[1 \ 0]^T$ (b) $\Phi_{div}(\mathbf{x}, 0)[0 \ 1]^T$ based on the Gaussian radial kernel.

$$\underbrace{\begin{bmatrix} \Phi_{div}(\mathbf{y}_1, \mathbf{y}_1) & \Phi_{div}(\mathbf{y}_1, \mathbf{y}_2) & \cdots & \Phi_{div}(\mathbf{y}_1, \mathbf{y}_n) \\ \Phi_{div}(\mathbf{y}_2, \mathbf{y}_1) & \Phi_{div}(\mathbf{y}_2, \mathbf{y}_2) & \cdots & \Phi_{div}(\mathbf{y}_2, \mathbf{y}_n) \\ \vdots & \vdots & \ddots & \vdots \\ \Phi_{div}(\mathbf{y}_n, \mathbf{y}_1) & \Phi_{div}(\mathbf{y}_n, \mathbf{y}_2) & \cdots & \Phi_{div}(\mathbf{y}_n, \mathbf{y}_n) \end{bmatrix}}_{A_{Y, \Phi_{div}}} \underbrace{\begin{bmatrix} \mathbf{c}_1 \\ \mathbf{c}_2 \\ \vdots \\ \mathbf{c}_n \end{bmatrix}}_{\underline{\mathbf{c}}} = \underbrace{\begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \vdots \\ \mathbf{u}_n \end{bmatrix}}_{\underline{\mathbf{u}}}. \quad (1.10)$$

Due to the symmetric structure of Φ_{div} , the matrix $A_{Y, \Phi_{div}}$ is also symmetric. It can also be shown to be positive definite for appropriately chosen ϕ [11], such as those in Table 1.1. This guarantees that (1.10) has a unique solution. Figure 1.4 shows a divergence-free vector field sampled at distinct points and the resulting divergence-free RBF interpolant.

The curl-free matrix-valued kernels are developed in a similar manner as the divergence-free ones. As before, we let ϕ be any scalar-valued radial kernel that is twice continuously differentiable and act on it with the appropriate differential operator. We define a curl-free matrix-valued kernels as [11]

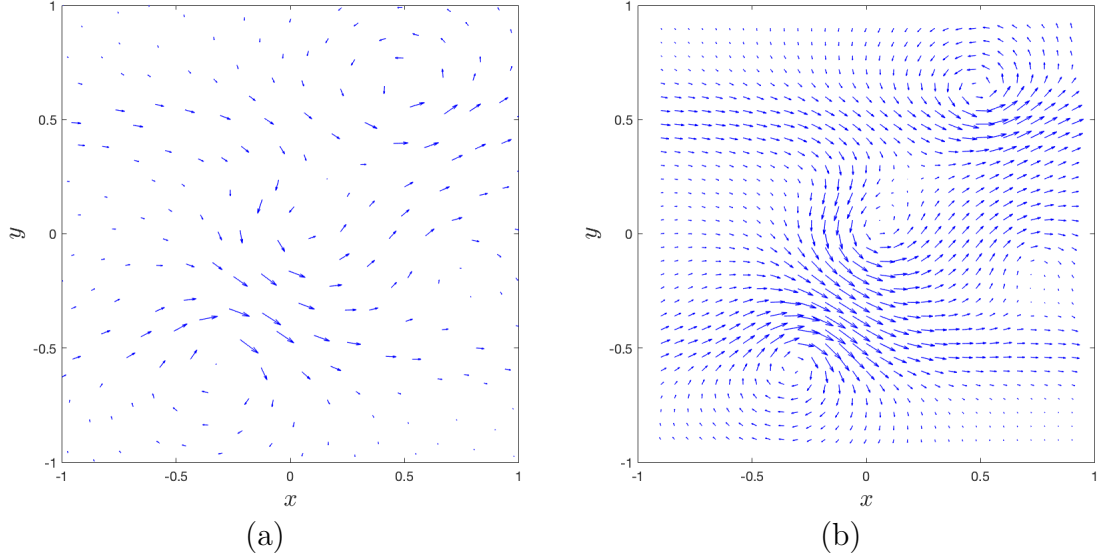


Figure 1.4: (a) The samples of a divergence-free vector field and (b) the interpolant of the field using the Gaussian kernel with $\varepsilon = 4.5$.

$$\Phi_{curl}(\mathbf{x}, \mathbf{y}) = -\nabla \nabla^T \phi(\mathbf{x}, \mathbf{y}). \quad (1.11)$$

As in the divergence-free case, we utilize the standard basis vectors $\mathbf{e}_j \in \mathbb{R}^d$ to show that the columns of this kernel are curl-free. The j^{th} column of Φ_{curl} is given by

$$\Phi_{curl}(\mathbf{x}, \mathbf{y}) \mathbf{e}_j = -\nabla \nabla^T \phi(\mathbf{x}, \mathbf{y}) \mathbf{e}_j = \nabla \left(-\nabla^T (\phi(\mathbf{x}, \mathbf{y}) \mathbf{e}_j) \right) = \nabla g, \quad (1.12)$$

where $g = -\partial \phi / \partial x^{(j)}$, and $x^{(j)}$ refers to the j^{th} coordinate of \mathbf{x} . Since g is a scalar function, we see that each column of Φ_{curl} is the gradient of a scalar, so they are curl-free; see Figure 1.5 for an illustration of the columns of Φ_{curl} in \mathbb{R}^2 . With this established, the curl-free vector RBF interpolant is given as

$$\mathbf{s}(\mathbf{x}) = \sum_{j=1}^n \Phi_{curl}(\mathbf{x}, \mathbf{y}_j) \mathbf{c}_j, \quad (1.13)$$

where the interpolation coefficients \mathbf{c}_j are found by solving the linear system as in (1.10), but with the matrix $A_{Y, \Phi_{curl}}$, whose $(j, k)^{th}$ $d \times d$ block is given by $\Phi_{curl}(\mathbf{y}_j, \mathbf{y}_k)$. Similar to the divergence-free case, Φ_{curl} is symmetric, which means the matrix

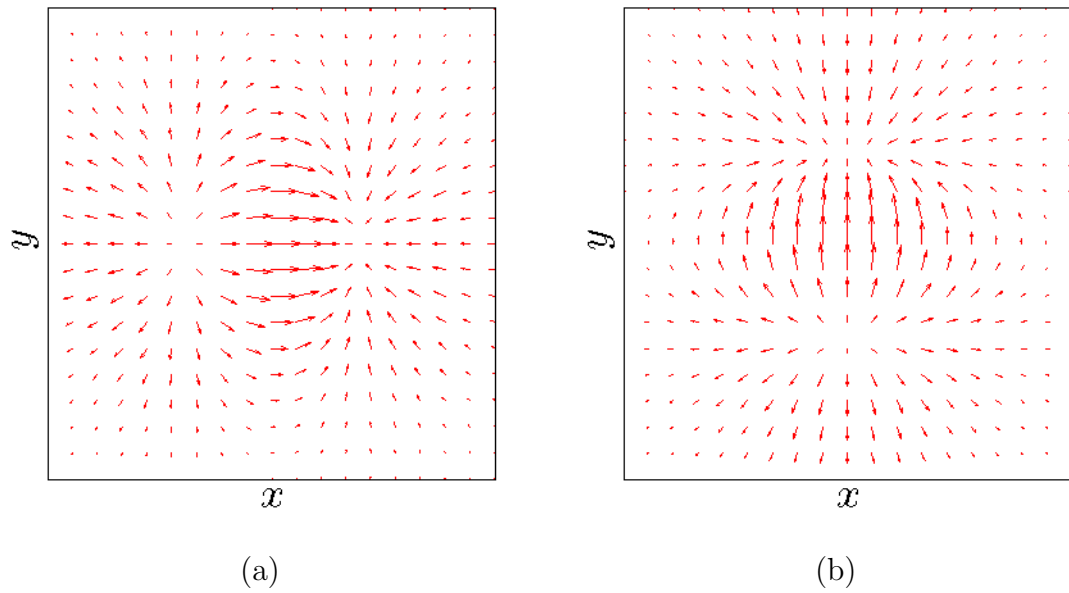


Figure 1.5: The columns of a curl-free kernel: (a) $\Phi_{curl}(\mathbf{x}, 0)[1 \ 0]^T$ (b) $\Phi_{cur}(\mathbf{x}, 0)[0 \ 1]^T$ based on the Gaussian radial kernel.

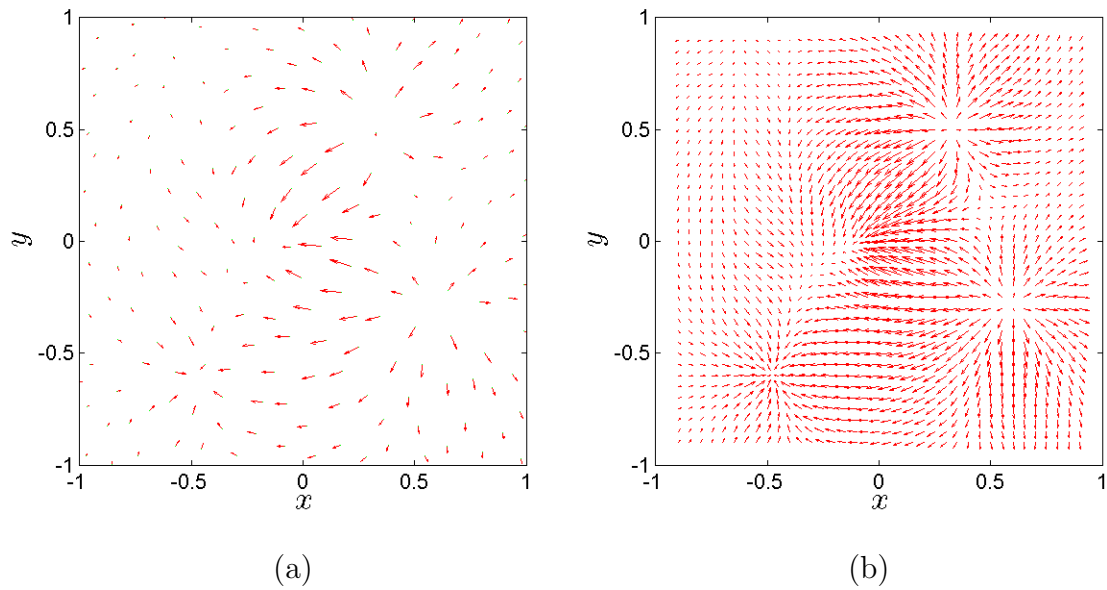


Figure 1.6: (a) The samples of a curl-free vector field and (b) the interpolant of the field using the Gaussian kernel with $\varepsilon = 4.5$.

$A_{Y, \Phi_{curl}}$ is also symmetric. $A_{Y, \Phi_{curl}}$ can also be shown to be positive definite for appropriately chosen ϕ , like those listed in Table 1.1 [11]. Figure 1.6 shows a curl-free vector field sampled at distinct points and the resulting curl-free RBF interpolant.

1.3 RBF Interpolation of Surface Divergence-Free and Curl-Free fields on the Sphere

The results from Section 1.2.2 dealt with the interpolation of divergence-free or curl-free vector fields in \mathbb{R}^d , but there are many applications, particularly in geophysics, where vector fields tangent to the surface of the sphere arise. For example, in the atmospheric sciences, horizontal wind fields are modeled as tangent vector fields, while the same is true of surface ocean currents in the oceanic sciences. In this section, we discuss how RBFs can be further customized for surface divergence-free or curl-free interpolation on the domain of the sphere. We denote the matrix-valued kernels used for defining these interpolants as Ψ_{div} and Ψ_{curl} , respectively. Note that while these kernels are the respective spherical analogues of Φ_{div} and Φ_{curl} , we cannot simply restrict Φ_{div} and Φ_{curl} to the surface of the sphere because this would not result in surface divergence-free/curl-free kernels.

1.3.1 Surface Differential Operators for Vector Fields in \mathbb{R}^3

In order to aid our discussion in subsequent sections, we will define tangential differential operators for vector fields on the two-sphere, \mathbb{S}^2 . Since \mathbb{S}^2 is a two-dimensional domain, we can define the (surface-) curl of a scalar function analogously to the curl of a scalar function in \mathbb{R}^2 , where curl should be understood as $\mathbf{n} \times \nabla$. The surface-curl of a scalar-valued function $f : \mathbb{S}^2 \rightarrow \mathbb{R}$, expressed in Cartesian coordinates, is given

as $Q_{\mathbf{x}}\nabla_{\mathbf{x}}f$, where $\mathbf{x} \in \mathbb{S}^2$, $\nabla_{\mathbf{x}}$ is the usual gradient on \mathbb{R}^3 applied to \mathbf{x} , and $Q_{\mathbf{x}}$ is the matrix that represents the cross product with the normal vector, i.e.

$$Q_{\mathbf{x}} := \begin{pmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{pmatrix}. \quad (1.14)$$

Additionally, the surface-gradient of a scalar-valued function $f : \mathbb{S}^2 \rightarrow \mathbb{R}$, expressed in Cartesian coordinates, is given as $P_{\mathbf{x}}\nabla_{\mathbf{x}}f$, where $P_{\mathbf{x}}$ projects vectors onto the tangent space on \mathbb{S}^2 at \mathbf{x} :

$$P_{\mathbf{x}} := I - \mathbf{x}\mathbf{x}^T = \begin{pmatrix} 1 - x^2 & -xy & -xz \\ -xy & 1 - y^2 & -yz \\ -xz & -yz & 1 - z^2 \end{pmatrix}. \quad (1.15)$$

Both the surface-curl and surface-gradient operators produce vector fields that are tangent to \mathbb{S}^2 at \mathbf{x} and are expressed with respect to the standard Cartesian coordinate basis. We also note here that the surface-curl of a scalar function on \mathbb{S}^2 is divergence-free, and fields that are surface-gradients of scalar functions on \mathbb{S}^2 are surface curl-free [13]. With these surface operators defined, we can now discuss vector RBF interpolation on the surface of the sphere.

1.3.2 Vector RBF Interpolation on the Sphere

We will begin with the derivation for the surface divergence-free matrix-valued RBF kernel for vector fields tangent to the sphere, which was first developed by Narcowich, Ward, and Wright [23]. Note that we will use extrinsic (Cartesian) coordinates because they do not suffer from pole singularities, unlike surface-based coordinate systems on the sphere. Let $\mathbf{x}, \mathbf{y} \in \mathbb{S}^2$ and consider a scalar-valued radial kernel centered at \mathbf{y} , $\phi(\|\mathbf{x} - \mathbf{y}\|)$. We then construct the 3-by-3 matrix-valued kernel, Ψ_{div} , using the surface-curl operator from Section 1.3.1 as follows:

$$\begin{aligned}
\Psi_{div}(\mathbf{x}, \mathbf{y}) &= (Q_{\mathbf{x}} \nabla_{\mathbf{x}}) (Q_{\mathbf{y}} \nabla_{\mathbf{y}})^T \phi(\|\mathbf{x} - \mathbf{y}\|) \\
&= Q_{\mathbf{x}} (\nabla_{\mathbf{x}} \nabla_{\mathbf{y}}^T \phi(\|\mathbf{x} - \mathbf{y}\|)) Q_{\mathbf{y}}.
\end{aligned} \tag{1.16}$$

Notice that the matrix $Q_{\mathbf{y}}$ eliminates the normal component of a vector, so for any $\mathbf{c} = (c_1, c_2, c_3)^T$, $\Psi_{div}(\mathbf{x}, \mathbf{y})\mathbf{c}$ is tangent to \mathbb{S}^2 at \mathbf{x} . Furthermore

$$\begin{aligned}
\Psi_{div}(\mathbf{x}, \mathbf{y})\mathbf{c} &= [Q_{\mathbf{x}} (\nabla_{\mathbf{x}} \nabla_{\mathbf{y}}^T \phi(\|\mathbf{x} - \mathbf{y}\|)) Q_{\mathbf{y}}] \mathbf{c} \\
&= Q_{\mathbf{x}} \nabla_{\mathbf{x}} \underbrace{[\nabla_{\mathbf{y}}^T (\phi(\|\mathbf{x} - \mathbf{y}\|)) Q_{\mathbf{y}} \mathbf{c}]}_f \\
&= Q_{\mathbf{x}} (\nabla_{\mathbf{x}} f),
\end{aligned} \tag{1.17}$$

where f is a scalar-valued function. Since $\Psi_{div}(\mathbf{x}, \mathbf{y})\mathbf{c}$ is equivalent to the surface-curl of a scalar function, we know that it is surface divergence-free. With this kernel, we can now construct an interpolant to a divergence-free tangent vector field on \mathbb{S}^2 .

Similar to the interpolation process described in Section 1.2.2, we begin with distinct nodes $Y = \{\mathbf{y}_j\}_{j=1}^n = \{(x_j, y_j, z_j)\}_{j=1}^n \subset \mathbb{S}^2$ and a surface divergence-free tangent vector field \mathbf{f} sampled on Y , $\{\mathbf{f}_j\}_{j=1}^n = \{[f_{j,1} f_{j,2} f_{j,3}]^T\}_{j=1}^n$; see Figure 1.8 (a) for an illustration. The surface divergence-free RBF interpolant takes the form

$$\mathbf{t}(\mathbf{x}) = \sum_{j=1}^n \Psi_{div}(\mathbf{x}, \mathbf{y}_j) \mathbf{c}_j, \tag{1.18}$$

where the interpolation coefficients \mathbf{c}_j are tangent to \mathbb{S}^2 at \mathbf{y}_j . This assumption is needed to make the interpolation problem well-posed. We note here that solving for these interpolation coefficient vectors is a two-dimensional problem since \mathbf{f}_j are really two-dimensional vectors, since they can be written as a sum of two orthonormal vectors. This creates an issue due to the terms in the sum (1.18) being three-dimensional vectors. A naïve approach to solving for the \mathbf{c}_j 's will lead to a singular system of equations. Therefore we will explain how to set up the vector interpolant so that the corresponding matrix for determining the interpolation coefficient vectors is non-singular.

We denote an orthonormal coordinate system at each node, \mathbf{y}_j , as $\{\mathbf{d}_j, \mathbf{e}_j, \mathbf{n}_j\}$, where \mathbf{n}_j is the outward normal to \mathbb{S}^2 , \mathbf{e}_j is a unit tangent vector, and $\mathbf{d}_j = \mathbf{n}_j \times \mathbf{e}_j$. Then we see that $\mathbf{n}_j = \mathbf{y}_j$ (since on the unit sphere the outward normal at \mathbf{y}_j is just \mathbf{y}_j) and choose \mathbf{d}_j and \mathbf{e}_j to be the standard meridional and zonal vectors, respectively:

$$\mathbf{d}_j = \frac{1}{\sqrt{1-z_j^2}} \begin{bmatrix} -z_j x_j \\ -z_j y_j \\ 1-z_j^2 \end{bmatrix}, \quad \mathbf{e}_j = \frac{1}{\sqrt{1-z_j^2}} \begin{bmatrix} -y_j \\ x_j \\ 0 \end{bmatrix}. \quad (1.19)$$

It is important to note here that \mathbf{d}_j and \mathbf{e}_j form an orthonormal basis for the tangent space of \mathbb{S}^2 at \mathbf{y}_j . Additionally, if $\mathbf{y} = [0, 0, 1]$ or $\mathbf{y} = [0, 0, -1]$, e.g. at one of the poles, we can pick any orthogonal vectors in the plane $z = 1$ or $z = -1$, respectively. Then the surface divergence-free vector RBF interpolant to the samples of \mathbf{f} is constructed from linear combinations of the tangent vector basis $\{(\Psi_{div}(\mathbf{x}, \mathbf{y}_j)\mathbf{d}_j, \Psi_{div}(\mathbf{x}, \mathbf{y}_j)\mathbf{e}_j)\}_{j=1}^n$, i.e. the interpolant is of the form

$$\mathbf{t}(\mathbf{x}) = \sum_{j=1}^n \Psi_{div}(\mathbf{x}, \mathbf{y}_j) \underbrace{[\alpha_j \mathbf{d}_j + \beta_j \mathbf{e}_j]}_{\mathbf{c}_j}, \quad (1.20)$$

where the unknowns α_j and β_j are determined by solving $\mathbf{t}(\mathbf{y}_i) = \mathbf{f}_i$, $i = 1, \dots, n$. Illustrations of the zonal and meridional basis vectors formed from Ψ_{div} are displayed in Figure 1.7.

Now solving for the interpolation coefficient vectors \mathbf{c}_j in (1.20) is equivalent to solving the linear system of equations,

$$\sum_{j=1}^n \Psi_{div}(\mathbf{y}_i, \mathbf{y}_j) [\alpha_j \mathbf{d}_j + \beta_j \mathbf{e}_j] = \mathbf{f}_i, \quad 1 \leq i \leq n, \quad (1.21)$$

for α_j and β_j . However, since \mathbf{f}_i has three components, (1.21) is not a square system. We note that we can make (1.21) a square system by expressing \mathbf{f}_i in terms of \mathbf{d}_i and \mathbf{e}_i as $\mathbf{f}_i = \gamma_i \mathbf{d}_i + \delta_i \mathbf{e}_i$, where

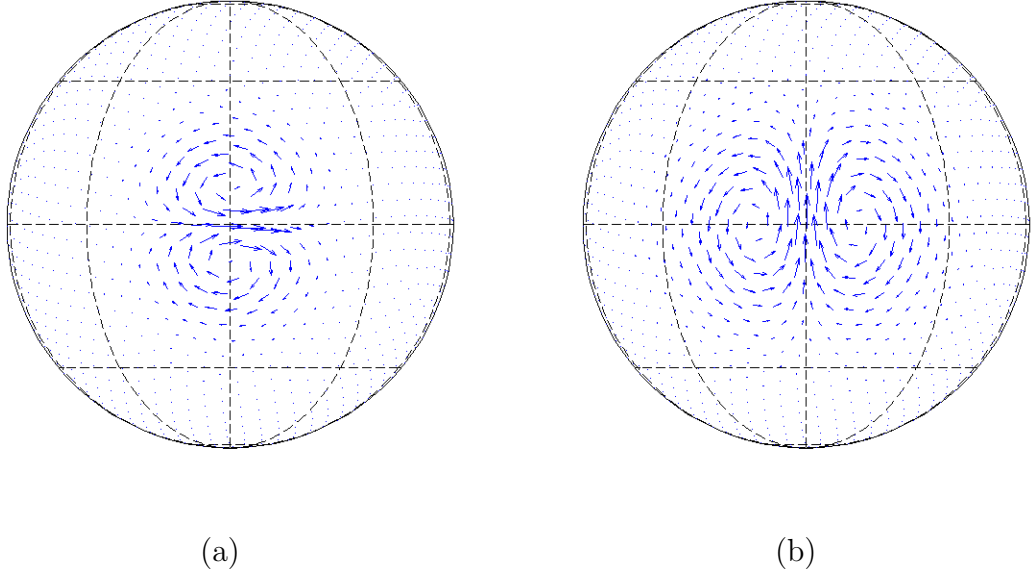


Figure 1.7: The two components of the tangent vector basis at \mathbf{y}_j : (a) Zonal basis, $\Psi_{div}(\mathbf{x}, \mathbf{y}_j)\mathbf{e}_j$ (b) Meridional basis, $\Psi_{div}(\mathbf{x}, \mathbf{y}_j)\mathbf{d}_j$

$$\begin{bmatrix} \gamma_i \\ \delta_i \end{bmatrix} = \begin{bmatrix} \mathbf{d}_i^T \\ \mathbf{e}_i^T \end{bmatrix} \mathbf{f}_i. \quad (1.22)$$

Using this, we can rewrite (1.21) as the $2n$ -by- $2n$ linear system

$$\sum_{j=1}^n \underbrace{\begin{pmatrix} \begin{bmatrix} \mathbf{d}_i^T \\ \mathbf{e}_i^T \end{bmatrix} \Psi_{div}(\mathbf{y}_i, \mathbf{y}_j) \begin{bmatrix} \mathbf{d}_j & \mathbf{e}_j \end{bmatrix} \end{pmatrix}}_{A_{\Psi_{div},(i,j)}} \begin{bmatrix} \alpha_j \\ \beta_j \end{bmatrix} = \begin{bmatrix} \gamma_i \\ \delta_i \end{bmatrix}, \quad 1 \leq i \leq n. \quad (1.23)$$

The $(i, j)^{\text{th}}$ 2-by-2 block of this interpolation matrix is denoted by $A_{\Psi_{div},(i,j)}$ and is given explicitly as [13]:

$$A_{\Psi_{div},(i,j)} = \begin{bmatrix} -\mathbf{e}_i \cdot \mathbf{e}_j & \mathbf{e}_i \cdot \mathbf{d}_j \\ \mathbf{d}_i \cdot \mathbf{e}_j & -\mathbf{d}_i \cdot \mathbf{d}_j \end{bmatrix} \eta(r_{ij}) + \begin{bmatrix} \mathbf{e}_i \cdot \mathbf{n}_j \\ -\mathbf{d}_i \cdot \mathbf{n}_j \end{bmatrix} \begin{bmatrix} \mathbf{n}_i \cdot \mathbf{e}_j & -\mathbf{n}_i \cdot \mathbf{d}_j \end{bmatrix} \zeta(r_{ij}), \quad (1.24)$$

where $r_{ij} = \|\mathbf{y}_i - \mathbf{y}_j\|$, $\eta(r) = \phi'(r)/r$, and $\zeta(r) = \eta'(r)/r$. The interpolation matrix that arises from these entries is of size $2n$ -by- $2n$ and is positive definite (and thus, invertible) if Ψ_{div} is constructed from any of the scalar kernels in Table 1.1 [23].

Figure 1.8 (b) illustrates an interpolated surface divergence-free vector field on \mathbb{S}^2

using the surface divergence-free RBF interpolant.

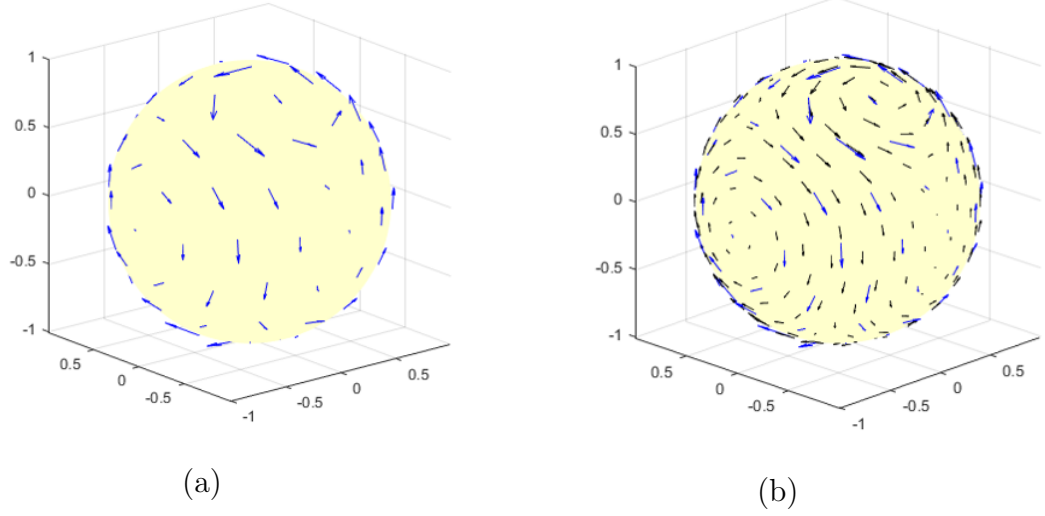


Figure 1.8: (a) The scattered samples of a surface divergence-free vector field in blue and (b) the interpolant of the field using the surface divergence-free RBF interpolant in black.

We note that the construction of a surface curl-free interpolant using Ψ_{curl} for scattered samples of a surface curl-free field is similar to that of the divergence-free process. So we discuss the curl-free case less extensively, highlighting the differences from the divergence-free case.

As before, we let $\mathbf{x}, \mathbf{y} \in \mathbb{S}^2$ and consider the scalar-valued radial kernel centered at \mathbf{y} , $\phi(\|\mathbf{x} - \mathbf{y}\|)$. We then construct the 3-by-3 matrix-valued kernel, Ψ_{curl} , using the surface-gradient operator from Section 1.3.1 as follows:

$$\begin{aligned} \Psi_{curl}(\mathbf{x}, \mathbf{y}) &= (P_{\mathbf{x}} \nabla_{\mathbf{x}}) (P_{\mathbf{y}} \nabla_{\mathbf{y}})^T \phi(\|\mathbf{x} - \mathbf{y}\|) \\ &= -P_{\mathbf{x}} (\nabla_{\mathbf{x}} \nabla_{\mathbf{y}}^T \phi(\|\mathbf{x} - \mathbf{y}\|)) P_{\mathbf{y}}. \end{aligned} \tag{1.25}$$

Again due to the operator $P_{\mathbf{y}}$ acting on a vector $\mathbf{c} = (c_1, c_2, c_3)^T$, $\Psi_{curl}(\mathbf{x}, \mathbf{y})\mathbf{c}$ is tangent to \mathbb{S}^2 at \mathbf{x} . Furthermore,

$$\begin{aligned}
\Psi_{curl}(\mathbf{x}, \mathbf{y})\mathbf{c} &= [-P_{\mathbf{x}} (\nabla_{\mathbf{x}} \nabla_{\mathbf{y}}^T \phi(\|\mathbf{x} - \mathbf{y}\|)) P_{\mathbf{y}}] \mathbf{c} \\
&= P_{\mathbf{x}} \nabla_{\mathbf{x}} \underbrace{[-\nabla_{\mathbf{y}}^T (\phi(\|\mathbf{x} - \mathbf{y}\|)) P_{\mathbf{y}} \mathbf{c}]}_g \\
&= P_{\mathbf{x}} (\nabla_{\mathbf{x}} g),
\end{aligned} \tag{1.26}$$

where g is a scalar-valued function. Since $\Psi_{curl}(\mathbf{x}, \mathbf{y})\mathbf{c}$ is equivalent to the surface gradient of a scalar function, we know that it is surface curl-free. With this kernel, we can now construct an interpolant to a surface curl-free tangent vector field on \mathbb{S}^2 .

As before, we begin with distinct nodes $Y = \{\mathbf{y}_j\}_{j=1}^n = \{(x_j, y_j, z_j)\}_{j=1}^n \subset \mathbb{S}^2$, but now we have a surface curl-free tangent vector field \mathbf{f} sampled on Y , $\{\mathbf{f}_j\}_{j=1}^n = \{[f_{j,1} f_{j,2} f_{j,3}]^T\}_{j=1}^n$; see Figure 1.10 for an illustration. The surface curl-free RBF interpolant takes the form

$$\mathbf{t}(\mathbf{x}) = \sum_{j=1}^n \Psi_{curl}(\mathbf{x}, \mathbf{y}_j) \mathbf{c}_j, \tag{1.27}$$

where \mathbf{c}_j is tangent to \mathbb{S}^2 at \mathbf{y}_j . We make the same modification as in (1.20) but with

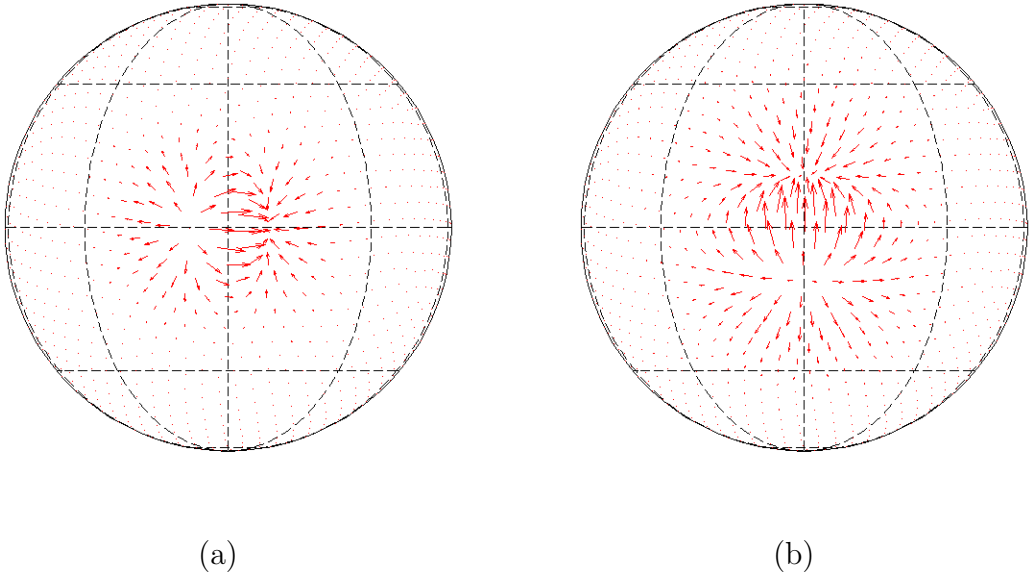


Figure 1.9: The two components of the tangent vector basis at \mathbf{y}_j : (a) Zonal basis, $\Psi_{curl}(\mathbf{x}, \mathbf{y}_j)\mathbf{e}_j$ (b) Meridional basis, $\Psi_{curl}(\mathbf{x}, \mathbf{y}_j)\mathbf{d}_j$

the tangent vector basis $\{(\Psi_{curl}(\mathbf{x}, \mathbf{y}_j)\mathbf{d}_j, \Psi_{curl}(\mathbf{x}, \mathbf{y}_j)\mathbf{e}_j)\}_{j=1}^n$ in order to get the $2n$ -by- $2n$ linear system for determining the interpolation coefficients $c_j = [\alpha_j \mathbf{e}_j \quad \beta_j \mathbf{d}_j]$:

$$\sum_{j=1}^n \underbrace{\begin{pmatrix} [\mathbf{d}_i^T] \\ \Psi_{curl}(\mathbf{y}_i, \mathbf{y}_j) [\mathbf{d}_j \quad \mathbf{e}_j] \end{pmatrix}}_{A_{\Psi_{curl},(i,j)}} \begin{bmatrix} \alpha_j \\ \beta_j \end{bmatrix} = \begin{bmatrix} \gamma_i \\ \delta_i \end{bmatrix}, \quad 1 \leq i \leq n. \quad (1.28)$$

Illustrations of the zonal and meridional basis vectors formed from Ψ_{curl} are displayed in Figure 1.9. The $(i, j)^{\text{th}}$ 2-by-2 block of this interpolation matrix is denoted by $A_{\Psi_{curl},(i,j)}$, and is given explicitly as [13]:

$$A_{\Psi_{curl},(i,j)} = \begin{bmatrix} \mathbf{d}_i \cdot \mathbf{d}_j & \mathbf{d}_i \cdot \mathbf{e}_j \\ \mathbf{e}_i \cdot \mathbf{d}_j & \mathbf{e}_i \cdot \mathbf{e}_j \end{bmatrix} \eta(r_{ij}) + \begin{bmatrix} \mathbf{d}_i \cdot \mathbf{n}_j \\ \mathbf{e}_i \cdot \mathbf{n}_j \end{bmatrix} [\mathbf{n}_i \cdot \mathbf{d}_j \quad \mathbf{n}_i \cdot \mathbf{e}_j] \zeta(r_{ij}), \quad (1.29)$$

where r_{ij} , $\eta(r)$, and $\zeta(r)$ are defined as in (1.24). The interpolation matrix that arises from these entries is size $2n$ -by- $2n$ and is positive definite (and thus, invertible) if Ψ_{curl} is constructed from any of the scalar kernels in Table 1.1 [13]. Figure 1.10 (b) illustrates an interpolated surface curl-free vector field on \mathbb{S}^2 using the surface curl-free RBF interpolant.

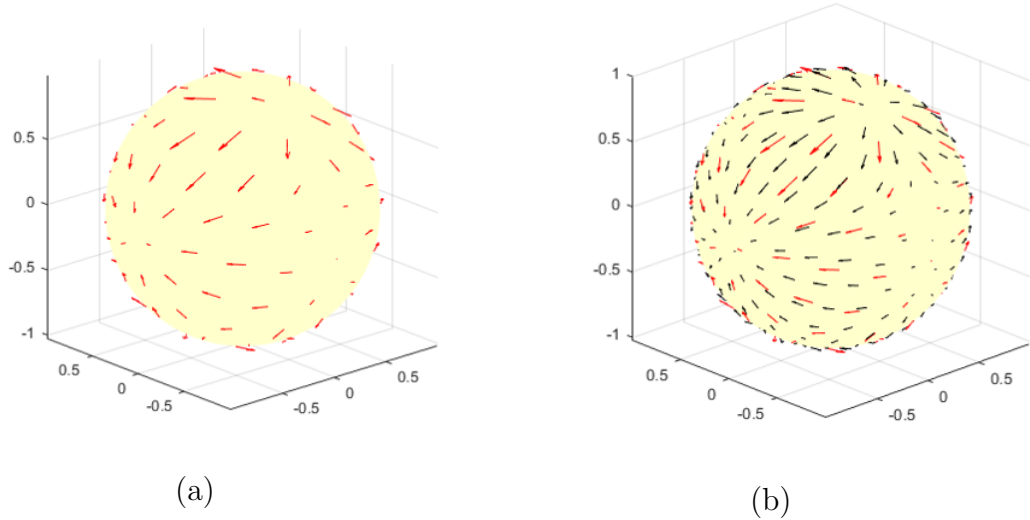


Figure 1.10: (a) The scattered samples of a surface curl-free vector field in red and (b) the interpolant of the field using the surface curl-free RBF interpolant in black.

We conclude by noting that according to the Helmholtz-Hodge decomposition, any vector field can be decomposed into a divergence-free component, a curl-free component, and a harmonic component. Since tangent vector fields on the sphere do not have harmonic components, we can decompose every tangent vector field on \mathbb{S}^2 uniquely into divergence-free and curl-free components. Fuselier and Wright introduced a technique for decomposing tangent vector fields on \mathbb{S}^2 using the Ψ_{div} and Ψ_{curl} kernels [13]. They demonstrated that the kernel for interpolating any tangent vector field on \mathbb{S}^2 is simply defined as $\Psi := \Psi_{div} + \Psi_{curl}$. Then given distinct nodes $Y = \{\mathbf{y}_j\}_{j=1}^n \subset \mathbb{S}^2$ and a surface tangent vector field \mathbf{f} sampled on Y , the interpolant is thus of the form

$$\mathbf{t}(\mathbf{x}) = \sum_{j=1}^n \Psi(\mathbf{x}, \mathbf{y}_j) \mathbf{c}_j = \underbrace{\sum_{j=1}^n \Psi_{div}(\mathbf{x}, \mathbf{y}_j) \mathbf{c}_j}_{\text{Div. free}} + \underbrace{\sum_{j=1}^n \Psi_{curl}(\mathbf{x}, \mathbf{y}_j) \mathbf{c}_j}_{\text{Curl free}}. \quad (1.30)$$

Fuselier and Wright furthermore showed that \mathbf{t} not only approximates the tangent vector field being interpolated, but also that the divergence-free and curl-free terms in the decomposition of \mathbf{t} approximate the corresponding parts of the underlying field [13].

1.4 Spherical Harmonics

In the algorithms presented in Chapters 2 and 3, we make heavy use of spherical harmonic expansions, both scalar and vector ones. We therefore give a brief introduction to these functions.

Spherical harmonics have many applications in the physical sciences, including computing atomic electron configurations, representing gravitational and magnetic fields of planetary bodies, and defining quantities of light transport in computer

graphics [3, 14, 30]. These expansions are the spherical analog of Fourier expansions, which can be used to represent functions defined on the unit circle. Since we are working specifically with functions on a sphere, we will be using scalar spherical harmonics and their vectorial analogue, vector spherical harmonics. The usefulness of the spherical harmonics for representing functions on the sphere is due in part to their inherent properties of orthogonality and completeness.

1.4.1 Scalar Spherical Harmonics

We denote the scalar spherical harmonic of degree $\mu \geq 0$ and order ν (see Figure 1.11 for illustrations) on \mathbb{S}^2 by Y_μ^ν . These functions are the eigenfunctions of the Laplace-Beltrami operator, which can be expressed in spherical coordinates on the unit sphere ($x = \sin \theta \cos \lambda$, $y = \sin \theta \sin \lambda$, $z = \cos \theta$) as

$$\nabla_{\mathbb{S}^2}^2 \equiv \frac{\partial^2}{\partial \theta^2} + \cot \theta \frac{\partial}{\partial \theta} + \frac{1}{\sin^2 \theta} \frac{\partial^2}{\partial \lambda^2}, \quad 0 \leq \theta \leq \pi, \quad -\pi \leq \lambda \leq \pi. \quad (1.31)$$

Each spherical harmonic satisfies $\nabla_{\mathbb{S}^2}^2 Y_\mu^\nu = -\mu(\mu + 1)Y_\mu^\nu$, and for each μ there are $2\mu + 1$ harmonics with the eigenvalue $-\mu(\mu + 1)$, enumerated by $-\mu \leq \nu \leq \mu$ [21]. This thesis will use the real-form of the spherical harmonics functions in Cartesian coordinates. For $(x, y, z) \in \mathbb{S}^2$, they are as follows

$$Y_\mu^\nu(x, y, z) = \begin{cases} \sqrt{\frac{2\mu+1}{4\pi}} \sqrt{\frac{(\mu-\nu)!}{(\mu+\nu)!}} P_\mu^\nu(z) \cos\left(\nu \tan^{-1}\left(\frac{y}{x}\right)\right), & \nu = 0, 1, \dots, \mu, \\ \sqrt{\frac{2\mu+1}{4\pi}} \sqrt{\frac{(\mu-\nu)!}{(\mu+\nu)!}} P_\mu^\nu(z) \sin\left(-\nu \tan^{-1}\left(\frac{y}{x}\right)\right), & \nu = -\mu, \dots, -1 \end{cases}. \quad (1.32)$$

Here $P_\mu^\nu(z)$ are the associated Legendre functions of degree μ and order ν . The spherical harmonics form a complete, orthonormal set of basis functions for the space of square-integrable functions on \mathbb{S}^2 , which we denote by $L^2(\mathbb{S}^2)$ [2]. Thus, any function $f \in L^2(\mathbb{S}^2)$ can be uniquely represented as

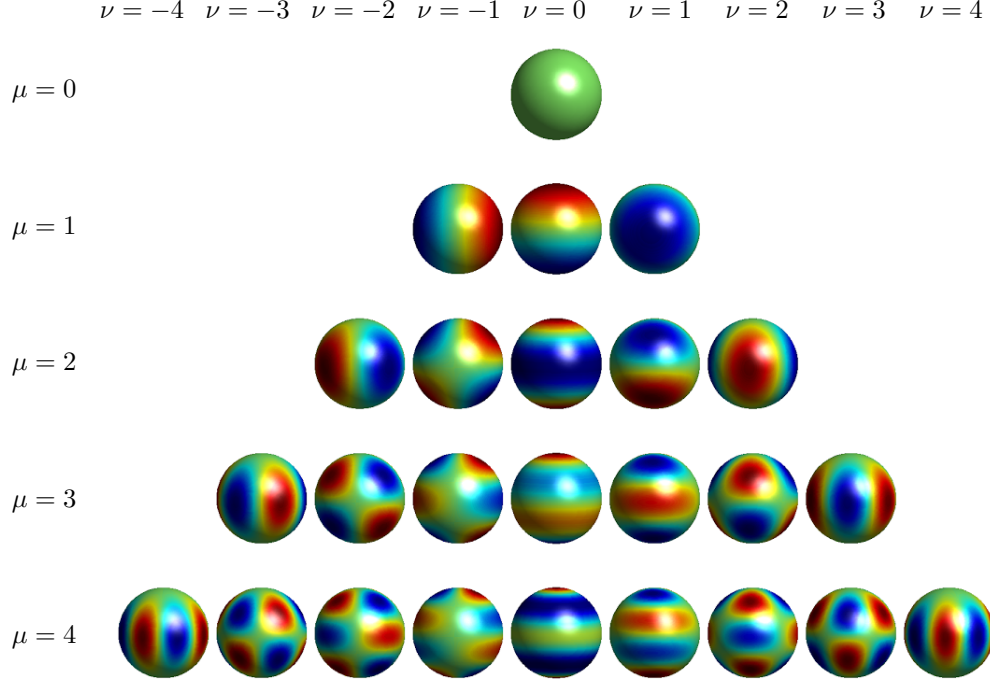


Figure 1.11: Pseudocolor plot of the scalar spherical harmonics basis functions of degrees $\mu = 0, 1, 2, 3, 4$ and orders $\nu = -\mu, \dots, \mu$. The colors range from blue to red, which correspond to negative and positive values, respectively.

$$f(x, y, z) = \sum_{\mu=0}^{\infty} \sum_{\nu=-\mu}^{\mu} c_{\mu,\nu} Y_{\mu}^{\nu}(x, y, z), \quad (1.33)$$

where $c_{\mu,\nu}$ are found using the usual L^2 -inner product for scalar functions on the sphere, $c_{\mu,\nu} = \langle f, Y_{\mu}^{\nu} \rangle$ [2].

1.4.2 Vector Spherical Harmonics

Vector spherical harmonics are the vectorial analogue of scalar spherical harmonics, and they are used for representing vector-valued functions on the sphere. There are three L^2 -orthogonal types of these functions: one type that is normal to the sphere, and two types that are tangent to the sphere [28]. In this thesis, we are working with vector fields that are tangent to the sphere. Therefore, we are interested in deriving the tangential vector spherical harmonics, which are separated into divergence-free

and curl-free terms. For the following derivations, we use the real-form of the scalar-valued spherical harmonic functions in Cartesian coordinates, as defined in (1.32).

We obtain the normalized surface divergence-free vector spherical harmonics by applying the surface-curl operator to the scalar spherical harmonic functions at \mathbf{x} , $Y_\mu^\nu(\mathbf{x})$:

$$\tilde{\mathbf{w}}_\mu^\nu = \frac{Q_{\mathbf{x}} \nabla_{\mathbf{x}} Y_\mu^\nu(\mathbf{x})}{\mu(\mu+1)}, \quad (1.34)$$

provided that $\mu \neq 0$ [28]. Since these are expressed as the surface-curl of scalar-valued functions, they are surface divergence-free. Similarly, we obtain the surface curl-free vector spherical harmonics by applying the surface-gradient to $Y_\mu^\nu(\mathbf{x})$:

$$\tilde{\mathbf{z}}_\mu^\nu = \frac{P_{\mathbf{x}} \nabla_{\mathbf{x}} Y_\mu^\nu}{\mu(\mu+1)}. \quad (1.35)$$

We see that these are surface curl-free since they are the surface-gradient of scalar-valued functions. They can also be shown to be orthonormal in $L^2(\mathbb{S}^2)$ [28]. We will denote the non-normalized surface divergence-free and curl-free vector spherical harmonics as \mathbf{w}_μ^ν and \mathbf{z}_μ^ν , respectively. The surface divergence-free and curl-free vector spherical harmonics are the eigenfunctions of the vector Laplace-Beltrami operator, which operates on vector fields tangent to \mathbb{S}^2 . Additionally, they form a complete orthonormal set of basis functions for the spectral representation of vector functions on \mathbb{S}^2 [28]. We denote this space again by $L^2(\mathbb{S}^2)$, but we define it using the inner product for vector functions \mathbf{f} and \mathbf{g} on \mathbb{S}^2 :

$$\langle \mathbf{f}, \mathbf{g} \rangle = \int_{\mathbb{S}^2} \mathbf{f}^T \mathbf{g} dS,$$

where the dot product is taken in local coordinates [28]. With this inner product, we define the vector spherical harmonic expansion of a vector function $\mathbf{f} \in L^2(\mathbb{S}^2)$ as

$$\mathbf{f} = \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \left(\hat{\mathbf{f}}_{div}(\mu, \nu) \mathbf{w}_{\mu}^{\nu} + \hat{\mathbf{f}}_{curl}(\mu, \nu) \mathbf{z}_{\mu}^{\nu} \right),$$

where

$$\hat{\mathbf{f}}_{div}(\mu, \nu) := \langle \mathbf{f}, \mathbf{w}_{\mu}^{\nu} \rangle \quad \text{and} \quad \hat{\mathbf{f}}_{curl}(\mu, \nu) := \langle \mathbf{f}, \mathbf{z}_{\mu}^{\nu} \rangle.$$

Note here that the outer sum excludes $\mu = 0$. This is due to the fact that the constant spherical harmonic term, $Y_0^0(\mathbf{x})$, is annihilated by the surface-curl and surface-gradient operators.

As discussed in Section 1.3.2, the interpolation problem on the sphere requires that we utilize the tangent basis vectors at $\mathbf{x} \in \mathbb{S}^2$, $\mathbf{d}_{\mathbf{x}}$ and $\mathbf{e}_{\mathbf{x}}$. Therefore, it is relevant to introduce the notation for the non-normalized surface divergence-free and curl-free vector spherical harmonics in terms of these basis vectors:

$$\begin{aligned} G_{\mu}^{\nu}(\mathbf{x}) &= \mathbf{d}_{\mathbf{x}}^T \cdot Q_{\mathbf{x}} \nabla_{\mathbf{x}} Y_{\mu}^{\nu}(\mathbf{x}) && \text{(meridional, divergence-free),} \\ H_{\mu}^{\nu}(\mathbf{x}) &= \mathbf{e}_{\mathbf{x}}^T \cdot Q_{\mathbf{x}} \nabla_{\mathbf{x}} Y_{\mu}^{\nu}(\mathbf{x}) && \text{(zonal, divergence-free),} \\ K_{\mu}^{\nu}(\mathbf{x}) &= \mathbf{d}_{\mathbf{x}}^T \cdot P_{\mathbf{x}} \nabla_{\mathbf{x}} Y_{\mu}^{\nu}(\mathbf{x}) && \text{(meridional, curl-free),} \\ L_{\mu}^{\nu}(\mathbf{x}) &= \mathbf{e}_{\mathbf{x}}^T \cdot P_{\mathbf{x}} \nabla_{\mathbf{x}} Y_{\mu}^{\nu}(\mathbf{x}) && \text{(zonal, curl-free).} \end{aligned}$$

1.5 Overview of the Thesis

As discussed in this chapter, RBF interpolation is an effective method for approximating scalar functions and vector fields given only scattered data. However, we see in Table 1.1 that the radial kernels used in this interpolation process are dependent on the shape parameter ε , which controls the peakedness of the kernels. The shape parameter is the focus of this thesis because of how it affects the accuracy of RBF approximations. Specifically, it has been observed that smaller values of ε result in better approximations to a point at which ill-conditioning enters into the interpolation

system (1.6). After this point as $\varepsilon \rightarrow 0$, the RBF Direct method becomes numerically unstable and the resulting approximations become highly inaccurate.

Fornberg and Piret developed an algorithm that bypasses the ill-conditioning of scalar RBF interpolation on the sphere in this flat limit [7]. Titled the RBF-QR algorithm, their work is the foundation on which we conducted the research of this thesis. Vector-valued RBF interpolants on the sphere have the same dependency on the shape parameter as scalar-valued interpolants, because of their direct relation to the scalar-valued radial kernels. In this thesis, we develop the first numerically stable algorithm for vector-valued RBF interpolation on the sphere in the flat limit.

The rest of the thesis is structured as follows. In Chapter 2 we give an extensive explanation of the Scalar RBF-QR algorithm of Fornberg and Piret [7], concluding with numerical results. In Chapter 3 we present the main result of the thesis, namely the Vector RBF-QR algorithm. As this is an extension of the Scalar RBF-QR algorithm, the Vector RBF-QR algorithm will be derived in a similar manner. Finally, we will end the thesis with numerical results from the Vector RBF-QR algorithm in Chapter 4, followed by conclusions in Chapter 5.

CHAPTER 2

THE RBF-QR ALGORITHM FOR STABLE SCALAR-VALUED RBF INTERPOLATION ON THE SPHERE

2.1 Scalar-Valued RBF Interpolation in the Flat Limit

As discussed in the previous chapter, RBFs are used in many disciplines for scattered data approximation on surfaces. Recall that the linear system in (1.6), used to solve for the interpolation coefficients, is guaranteed to be nonsingular for the $\phi(r)$ functions listed in Table 1.1. Researchers observed that the conditioning of the linear system (1.6) and the accuracy of the resulting interpolant (1.5) are greatly dependent on the shape parameter, ε . Specifically, they noted that (1.6) is well-conditioned for

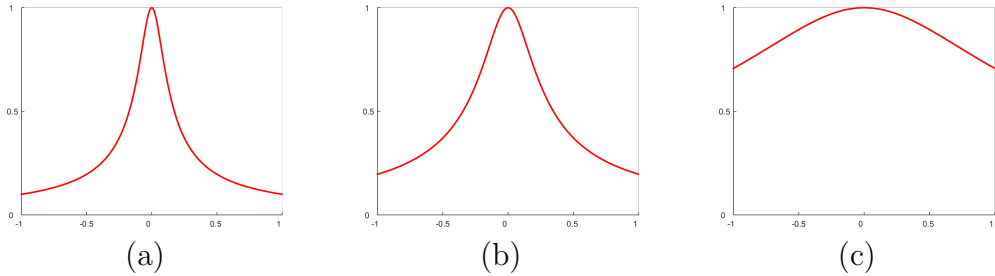


Figure 2.1: The inverse multiquadric kernel for (a) $\varepsilon = 10$, (b) $\varepsilon = 5$, and (c) $\varepsilon = 1$

large values of ε , but the interpolant gives a poor approximation of the underlying target function. This is due to the fact that ε controls the peakedness of the radial

kernels, where larger values of ε cause the functions to become more spiked. For example, as $\varepsilon \rightarrow \infty$ in the 1-D multiquadric function, the corresponding RBF interpolant converges to a piecewise linear interpolant. In contrast, the radial kernels become flatter and flatter as the shape parameter $\varepsilon \rightarrow 0$, hence the name “flat limit.” This is illustrated in Figure 2.1 for the inverse multiquadric function. Researchers also observed that for smaller values of ε , the interpolant (1.5) gives a better approximation of the underlying target function to a point at which ill-conditioning of the linear system (1.6) sets in. Figure 2.2 illustrates this phenomenon between ill-conditioning and accuracy for an interpolation problem on \mathbb{S}^2 .

There is extensive literature dedicated to finding the “optimal” shape parameter, i.e. the value of ε that results in the best approximation of the target function [4, 26]. However, the proposed methods are limited because of the disastrous ill-conditioning that enters the RBF Direct interpolation process in the flat limit. As researchers investigated the flat limit, they hypothesized that the error trend seen in Figure 2.2 would not increase rapidly, provided that this ill-conditioning was eliminated. The first step toward confirming this conjecture was recognizing where the ill-conditioning enters the problem. As $\varepsilon \rightarrow 0$, the basis functions all become 1, causing the linear system used to solve for the coefficients to become singular. In other words, the columns of A_Y become linearly dependent, and the condition number grows without bound, causing the coefficients to blow-up. They noticed that while the expansion coefficients blow-up as $\varepsilon \rightarrow 0$, the RBF interpolant remains well-behaved. In fact, Driscoll and Fornberg showed that for 1-D scattered data, the RBF interpolant converges to the Lagrange interpolating polynomial as $\varepsilon \rightarrow 0$ for all of the ϕ listed in Table 1.1 (and many others). They explained this convergence by first noting the interpolant (1.5), which we denote now by $s(\mathbf{x}, \varepsilon)$ in order to emphasize the

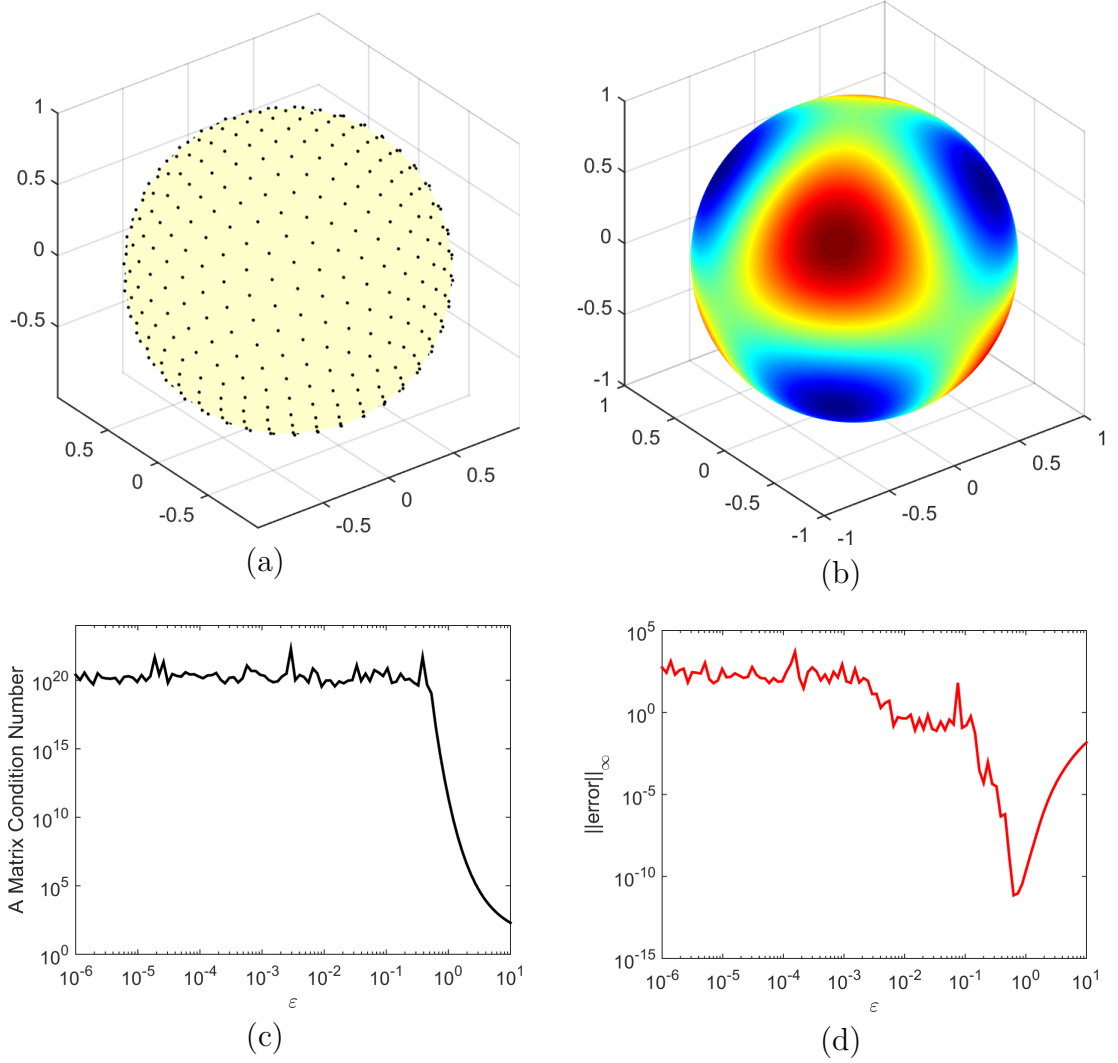


Figure 2.2: A problem illustrating ill-conditioning that enters the interpolation process in the RBF Direct method for an interpolation problem on the sphere consisting of (a) $n = 529$ quasi-uniformly distributed nodes and (b) the target function $f = \sin(xyz)$ on the sphere. (c) Condition number of the A_Y matrix in (1.6) vs ε . (d) Max norm error vs ε in the resulting RBF interpolant over the sphere computed with the RBF Direct approach. The IMQ kernel was used here.

dependence on ε , can be rewritten as

$$\begin{aligned} s(\mathbf{x}, \varepsilon) &= [\phi(\|\mathbf{x} - \mathbf{y}_1\|) \quad \phi(\|\mathbf{x} - \mathbf{y}_2\|) \quad \cdots \quad \phi(\|\mathbf{x} - \mathbf{y}_n\|)] \underline{c} \\ &= \underline{b}(\mathbf{x}, \varepsilon) A_Y^{-1}(\varepsilon) \underline{f}, \end{aligned} \quad (2.1)$$

where $A_Y(\varepsilon)$ now denotes the matrix in (1.6). They then showed that vast amounts

of cancellations occur when multiplying $\underline{b}(\mathbf{x}, \varepsilon)^T$ by $A_Y^{-1}(\varepsilon)$ that compensate for the divergence of the entries of $A_Y^{-1}(\varepsilon)$. In other words, computing \underline{c} directly via (1.6) and then from that the interpolant (1.5) is an ill-conditioned step in an otherwise well-conditioned interpolation process.

The first stable algorithm to bypass the inherent ill-conditioning of RBF Direct was the contour-Padé method, developed by Fornberg and Wright in 2004 [8] (see also [33]). Fornberg and Piret later developed a different stable algorithm for interpolation on the sphere [7], which they termed RBF-QR (see also the extensions to \mathbb{R}^2 and \mathbb{R}^3 [6]). This algorithm is the basis for the stable algorithm we develop in Chapter 3 for surface divergence-free and curl-free interpolation with RBFs. The remainder of this chapter gives an overview of the Scalar RBF-QR algorithm of Fornberg and Piret.

2.2 Scalar RBF-QR Algorithm

One of the ways to bypass the ill-conditioning of the scalar-valued RBF Direct interpolation system (1.6) in the flat limit is to replace the flat RBF basis with a well-conditioned one that spans the same space. By doing this, we get an equivalent interpolation result, but with a completely stable process. This is the key idea behind the Scalar RBF-QR method. Fornberg and Piret achieved this in their algorithm by first using scalar spherical harmonics to expand radial kernels. Then with some clever linear algebra, they were able to create a well-conditioned and equivalent basis.

2.2.1 Spherical Harmonic Expansion of RBF Kernels

In order to transform the RBF basis into a spherical harmonic basis, we can use the following formula (derived from the spherical harmonic addition theorem [2, 21]) for

the spherical harmonic expansion of each basis function [7]:

$$\phi(\|\mathbf{x} - \mathbf{y}_j\|) = \sum_{\mu=0}^{\infty} \sum_{\nu=-\mu}^{\mu'} \{c_{\mu,\varepsilon} \varepsilon^{2\mu} Y_{\mu}^{\nu}(\mathbf{y}_j)\} Y_{\mu}^{\nu}(\mathbf{x}), \quad (2.2)$$

where the symbol \sum' denotes that the $\nu = 0$ term is halved. Table 2.1 lists the expansion coefficients for many common radial kernels. These were first worked out by Hubbert and Baxter [18] for the radial kernels listed in Table 1.1. It is also important

Radial Kernel	Expansion Coefficient, $c_{\mu,\varepsilon}$
MQ	$\frac{-2\pi(2\varepsilon^2 + 1 + (\mu + 1/2)\sqrt{1 + 4\varepsilon^2})}{(\mu + 3/2)(\mu + 1/2)(\mu - 1/2)} \left(\frac{2}{1 + \sqrt{4\varepsilon^2 + 1}} \right)^{2\mu+1}$
IMQ	$\frac{4\pi}{(\mu + 1/2)} \left(\frac{2}{1 + \sqrt{4\varepsilon^2 + 1}} \right)^{2\mu+1}$
IQ	$\frac{4\pi^{3/2}\mu!}{\Gamma(\mu + \frac{3}{2})(1 + 4\varepsilon^2)^{\mu+1}} {}_2F_1(\mu + 1, \mu + 1; 2\mu + 2; \frac{4\varepsilon^2}{1 + 4\varepsilon^2})$
GA	$\frac{4\pi^{3/2}}{\varepsilon^{2\mu+1}} e^{-2\varepsilon^2} I_{\mu+1/2}(2\varepsilon^2)$

Table 2.1: SPH expansion coefficients for various radial kernels on the sphere. In the formula for the IQ kernel, ${}_2F_1(\dots)$ denotes the hypergeometric function, and in the formula for the GA kernel, $I_{\mu+1/2}$ denotes the Bessel function of the second kind. Note that the apparent singularity of the $c_{\mu,\varepsilon}$ for the GA kernel is a removable one due to the identity $\frac{I_{\mu+1/2}(2\varepsilon^2)}{\varepsilon^{2\mu+1}} = \frac{1}{\Gamma(\mu+1)\sqrt{\pi}} \int_{-1}^1 e^{2\varepsilon^2 t} (1 - t^2)^k dt$.

to mention that the coefficients listed in Table 2.1 can be calculated without the loss of any significant digits caused by numerical cancellations, even for vanishingly small ε [7].

We note that the expansion coefficients in Table 2.1 depend solely on μ , as opposed to those in (1.33), which depend on both μ and ν . This follows from the Funk-Hecke formula [2, 21]. In the next section, we will show that the Scalar RBF-QR algorithm avoids numerical underflow from the $\varepsilon^{2\mu}$ terms in (2.2) by performing matrix manipulations that introduce analytical cancellations of these powers of ε .

2.2.2 Matrix Representation and QR Factorization

Using the spherical harmonic expansion formula (2.2), we can rewrite each radial kernel in the RBF basis as an infinite sum:

$$\left\{ \begin{array}{l} \phi(\|\mathbf{x} - \mathbf{y}_1\|) = \frac{c_{0,\varepsilon}}{2} Y_0^0(\mathbf{y}_1) Y_0^0(\mathbf{x}) + \\ \quad + \varepsilon^2 c_{1,\varepsilon} \{Y_1^{-1}(\mathbf{y}_1) Y_1^{-1}(\mathbf{x}) + \frac{1}{2} Y_0^0(\mathbf{y}_1) Y_0^0(\mathbf{x}) + Y_1^1(\mathbf{y}_1) Y_1^1(\mathbf{x})\} + \\ \quad + \varepsilon^4 c_{2,\varepsilon} \{\dots\dots\dots\} + \varepsilon^6 c_{3,\varepsilon} \{\dots\dots\dots\} + \varepsilon^8 c_{4,\varepsilon} \{\dots\dots\dots\} + \dots \\ \phi(\|\mathbf{x} - \mathbf{y}_2\|) = \frac{c_{0,\varepsilon}}{2} Y_0^0(\mathbf{y}_2) Y_0^0(\mathbf{x}) + \\ \quad + \varepsilon^2 c_{1,\varepsilon} \{Y_1^{-1}(\mathbf{y}_2) Y_1^{-1}(\mathbf{x}) + \frac{1}{2} Y_0^0(\mathbf{y}_2) Y_0^0(\mathbf{x}) + Y_1^1(\mathbf{y}_2) Y_1^1(\mathbf{x})\} + \\ \quad + \varepsilon^4 c_{2,\varepsilon} \{\dots\dots\dots\} + \varepsilon^6 c_{3,\varepsilon} \{\dots\dots\dots\} + \varepsilon^8 c_{4,\varepsilon} \{\dots\dots\dots\} + \dots \\ \vdots \\ \phi(\|\mathbf{x} - \mathbf{y}_n\|) = \frac{c_{0,\varepsilon}}{2} Y_0^0(\mathbf{y}_n) Y_0^0(\mathbf{x}) + \\ \quad + \varepsilon^2 c_{1,\varepsilon} \{Y_1^{-1}(\mathbf{y}_n) Y_1^{-1}(\mathbf{x}) + \frac{1}{2} Y_0^0(\mathbf{y}_n) Y_0^0(\mathbf{x}) + Y_1^1(\mathbf{y}_n) Y_1^1(\mathbf{x})\} + \\ \quad + \varepsilon^4 c_{2,\varepsilon} \{\dots\dots\dots\} + \varepsilon^6 c_{3,\varepsilon} \{\dots\dots\dots\} + \varepsilon^8 c_{4,\varepsilon} \{\dots\dots\dots\} + \dots \end{array} \right. \quad (2.3)$$

We note here that the Scalar RBF-QR algorithm is easiest to describe and implement when the number of interpolation nodes, n , is a perfect square, i.e. $n = (\mu_0 + 1)^2$ for some $\mu_0 > 0$. This restriction allows for a unique way to split the RBF-QR system.

The summations in (2.3) are equivalent to the infinite matrix-vector product

$$\begin{aligned} & \begin{bmatrix} \phi(\|\mathbf{x} - \mathbf{y}_1\|) \\ \phi(\|\mathbf{x} - \mathbf{y}_2\|) \\ \vdots \\ \phi(\|\mathbf{x} - \mathbf{y}_n\|) \end{bmatrix} = \\ & = \begin{bmatrix} \frac{c_{0,\varepsilon}}{2} Y_0^0(\mathbf{y}_1) & \frac{c_{1,\varepsilon}}{1} Y_1^{-1}(\mathbf{y}_1) & \frac{c_{1,\varepsilon}}{2} Y_1^0(\mathbf{y}_1) & \frac{c_{1,\varepsilon}}{1} Y_1^1(\mathbf{y}_1) & \dots \\ \frac{c_{0,\varepsilon}}{2} Y_0^0(\mathbf{y}_2) & \frac{c_{1,\varepsilon}}{1} Y_1^{-1}(\mathbf{y}_2) & \frac{c_{1,\varepsilon}}{2} Y_1^0(\mathbf{y}_2) & \frac{c_{1,\varepsilon}}{1} Y_1^1(\mathbf{y}_2) & \dots \\ \vdots & \vdots & \vdots & \vdots & \dots \\ \frac{c_{0,\varepsilon}}{2} Y_0^0(\mathbf{y}_n) & \frac{c_{1,\varepsilon}}{1} Y_1^{-1}(\mathbf{y}_n) & \frac{c_{1,\varepsilon}}{2} Y_1^0(\mathbf{y}_n) & \frac{c_{1,\varepsilon}}{1} Y_1^1(\mathbf{y}_n) & \dots \end{bmatrix} \begin{bmatrix} \varepsilon^0 & & & & \\ & \varepsilon^2 & & & \\ & & \varepsilon^2 & & \\ & & & \varepsilon^2 & \\ & & & & \ddots \end{bmatrix} \begin{bmatrix} Y_0^0(\mathbf{x}) \\ Y_1^{-1}(\mathbf{x}) \\ Y_1^0(\mathbf{x}) \\ Y_1^1(\mathbf{x}) \\ \vdots \end{bmatrix} \\ & = B^\infty E^\infty Y^\infty. \end{aligned} \quad (2.4)$$

The first step of the Scalar RBF-QR algorithm truncates these infinite matrices based on the degree μ of the scalar spherical harmonics in order to make the method

computable. This truncation degree, $\mu = k$, must satisfy two conditions. First, k must be at least as large as the degree, μ_{trunc} , that would ensure each row on the left-hand side of (2.4) is approximated to machine precision by the corresponding row on the right-hand side. Second, the choice of k must also be at least as big as μ_0 , so that the matrix resulting from truncating B^∞ is at least square. Thus, we pick $k \geq \max[\mu_{trunc}, \mu_0]$. We denote the truncated system (2.4) as

$$\underbrace{\begin{bmatrix} \phi(\|\mathbf{x} - \mathbf{y}_1\|) \\ \phi(\|\mathbf{x} - \mathbf{y}_2\|) \\ \vdots \\ \phi(\|\mathbf{x} - \mathbf{y}_n\|) \end{bmatrix}}_{\tilde{p}} \approx \begin{bmatrix} & & & \\ & B & & \\ & & & \end{bmatrix} \begin{bmatrix} & & & \\ & E & & \\ & & & \end{bmatrix} \begin{bmatrix} Y \end{bmatrix}, \quad (2.5)$$

where B is now size n -by- m , E is size m -by- m , and Y is size m -by-1 for $m = (k+1)^2$.

Before describing the second step of the Scalar RBF-QR algorithm, we establish dimensions and structures of the matrices involved. The system (2.5) is of the form

$$\underbrace{\begin{bmatrix} B_0 & B_1 & \cdots & B_{\mu_0} & B_{\mu_0+1} & \cdots & B_k \end{bmatrix}}_B EY,$$

where B_μ , $0 \leq \mu \leq k$, are the block matrices of size n -by- $(2\mu+1)$ with entries

$$\{B_\mu\}_{i,j} = \begin{cases} c_{\mu,\varepsilon} Y_\mu^{j-(\mu+1)}(\mathbf{y}_i) & j \neq \mu+1, \\ \frac{c_{\mu,\varepsilon}}{2} Y_\mu^0(\mathbf{y}_i) & j = \mu+1, \end{cases} \quad j = 1, \dots, 2\mu+1, \quad i = 1, \dots, n.$$

The diagonal E matrix can be written as two square, diagonal blocks, E_1 and E_2

$$E = \left[\begin{array}{c|c} E_1 & \\ \hline & E_2 \end{array} \right],$$

where

$$E_1 = \begin{bmatrix} 1 & & & & \\ & \varepsilon^2 I_3 & & & \\ & & \varepsilon^4 I_5 & & \\ & & & \ddots & \\ & & & & \varepsilon^{2\mu_0} I_{2\mu_0+1} \end{bmatrix}, \quad (2.6)$$

$$E_2 = \begin{bmatrix} \varepsilon^{2\mu_0+2} I_{2\mu_0+3} & & & & \\ & \varepsilon^{2\mu_0+4} I_{2\mu_0+5} & & & \\ & & \ddots & & \\ & & & \ddots & \\ & & & & \varepsilon^{2k} I_{2k+1} \end{bmatrix}, \quad (2.7)$$

and I_μ is the identity matrix of size μ -by- μ . Due to the truncation and restriction on n , we see that E_1 is of size n -by- n , and E_2 is of size $(m - n)$ -by- $(m - n)$. Finally, the Y vector is given by

$$\begin{aligned} Y &= \left[Y_0 \mid Y_1 \mid \cdots \mid Y_{\mu_0} \mid Y_{\mu_0+1} \mid \cdots \mid Y_k \right]^T, \\ \{Y_\mu\}_j &= Y_\mu^{j-(\mu+1)}(\mathbf{x}), \quad j = 1, \dots, 2\mu + 1. \end{aligned}$$

The second step of the Scalar RBF-QR algorithm is to compute a QR factorization of the B matrix. It is important to note that the goal of this is to form new linear combinations of the existing basis functions so that the new basis will span the same space (we will address the well-conditioned nature of this new basis in the next section). A QR factorization is ideal for this goal because it operates on the columns of the matrix without combining terms in successive columns. In other words, the QR factorization will produce an upper triangular matrix that is composed of new linear combinations of the rows of B . This upper triangular matrix is exactly what the algorithm utilizes to create the new, stable basis. We let \tilde{p} denote the vector containing the elements of the RBF basis on the left-hand side of (2.5) so that the

QR-factorization of B leads to

$$\tilde{p} \approx Q \underbrace{[R_1 | R_2]}_R \left[\begin{array}{c|c} E_1 & \\ \hline & E_2 \end{array} \right] Y, \quad (2.8)$$

where R_1 is an n -by- n upper-triangular matrix, and R_2 is an n -by- $(m-n)$ full matrix. Note that in terms of powers of ε , all entries of R_1 and R_2 are $O(1)$ since these powers have been factored out into E .

The third step now proceeds by some clever factoring of the expression (2.8). Assuming that the diagonal entries of R_1 are non-zero so that it is invertible¹, we can re-write this system as

$$\tilde{p} \approx QR_1 [I_n | R_1^{-1}R_2] \left[\begin{array}{c|c} E_1 & \\ \hline & E_2 \end{array} \right] Y.$$

Exploiting the structure of E given in (2.6) and (2.7), we can re-write this expression again as

$$\begin{aligned} \tilde{p} &\approx QR_1 [E_1 | R_1^{-1}R_2E_2] Y \\ &= QR_1 E_1 \underbrace{[I_n | E_1^{-1}R_1^{-1}R_2E_2]}_{\tilde{B}} Y. \end{aligned} \quad (2.9)$$

It follows from this new expression that any element in the span of the rows of \tilde{p} can be represented to machine precision by a linear combination of the rows of $\tilde{B}Y$.

The fourth and final step of this algorithm is to reformulate \tilde{B} using properties of the Hadamard product of diagonal matrices. We will also use this to support the claim that this new basis is much better conditioned than the original for small ε , and thus much more suitable for numerical work. To begin the final step, we consider the product in the second block-column of \tilde{B} from (2.9). Using the properties of

¹This will be true if the nodes are unisolvent with respect to the spherical harmonic basis.

multiplication of a matrix on the left and right by diagonal matrices (see Appendix A), we have

$$E_1^{-1} R_1^{-1} R_2 E_2 = (R_1^{-1} R_2) \circ \underbrace{(E_1^{-1} J_{n,m-n} E_2)}_{\tilde{E}}, \quad (2.10)$$

where $J_{n,m-n}$ is the n -by- $(m-n)$ matrix whose entries are all 1, and \circ denotes the Hadamard product, or entry wise multiplication. After considering the structure of E_1 and E_2 given in (2.6) and (2.7), respectively, we see that the entries of \tilde{E} are given explicitly by

$$\tilde{E} = \left[\begin{array}{c|c|c|c} \varepsilon^{2\mu_0+2} J_{1,2\mu_0+3} & \varepsilon^{2\mu_0+4} J_{1,2\mu_0+5} & \cdots & \cdots & \varepsilon^{2k} J_{1,2k+1} \\ \hline \varepsilon^{2\mu_0} J_{3,2\mu_0+3} & \varepsilon^{2\mu_0+2} J_{3,2\mu_0+5} & \cdots & \cdots & \varepsilon^{2k-2} J_{3,2k+1} \\ \hline \varepsilon^{2\mu_0-2} J_{5,2\mu_0+3} & \varepsilon^{2\mu_0} J_{5,2\mu_0+5} & \cdots & \cdots & \varepsilon^{2k-4} J_{5,2k+1} \\ \hline \vdots & \vdots & \ddots & \ddots & \vdots \\ \hline \varepsilon^4 J_{2\mu_0-1,2\mu_0+3} & \varepsilon^6 J_{2\mu_0-1,2\mu_0+5} & \cdots & \cdots & \varepsilon^{2k-2\mu_0+2} J_{2\mu_0-1,2k+1} \\ \hline \varepsilon^2 J_{2\mu_0+1,2\mu_0+3} & \varepsilon^4 J_{2\mu_0+1,2\mu_0+5} & \cdots & \cdots & \varepsilon^{2k-2\mu_0} J_{2\mu_0+1,2k+1} \end{array} \right]. \quad (2.11)$$

Rewriting $\tilde{B}Y$ as

$$\tilde{B}Y = \left[I_n \mid (R_1^{-1} R_2) \circ \tilde{E} \right] Y,$$

and recalling $R_1^{-1} R_2$ is $O(1)$ in terms of the powers of ε , we see that each term in this new basis is now a spherical harmonic function with an $O(\varepsilon^2)$ perturbation. This follows from the property that the last block in the first column of (2.11) has an ε^2 term with the rest being ε^{2j} , $j \geq 2$. This shows that the new basis converges to the spherical harmonic basis as $\varepsilon \rightarrow 0$ and hence the RBF interpolant will converge to the spherical harmonic interpolant as $\varepsilon \rightarrow 0$ whenever the point set is unisolvent with respect to the spherical harmonics.

Note that with the above derivation, it is also possible to include the spherical harmonic coefficients $c_{\mu,\varepsilon}$ in the diagonal E_1 and E_2 matrices, and generate a similar analytical simplification for \tilde{E} . This has the added advantage of removing all ε de-

pendence in the actual QR numerical computation, and thus removing any numerical contamination for small ε . This approach unifies the Scalar RBF-QR method for all radial kernels since, for the same set of nodes, the only thing that would change is the \tilde{E} matrix.

2.2.3 Numerical Results

The numerical results from the Scalar RBF-QR algorithm confirmed the original hypothesis of researchers: with the ill-conditioning removed from the interpolation problem, the errors of the approximation did not blow up as $\varepsilon \rightarrow 0$. As a means of testing the algorithm, we present numerical results for the test problem described in Figure 2.2. Figure 2.2.3 shows the results of the RBF-QR algorithm on this test

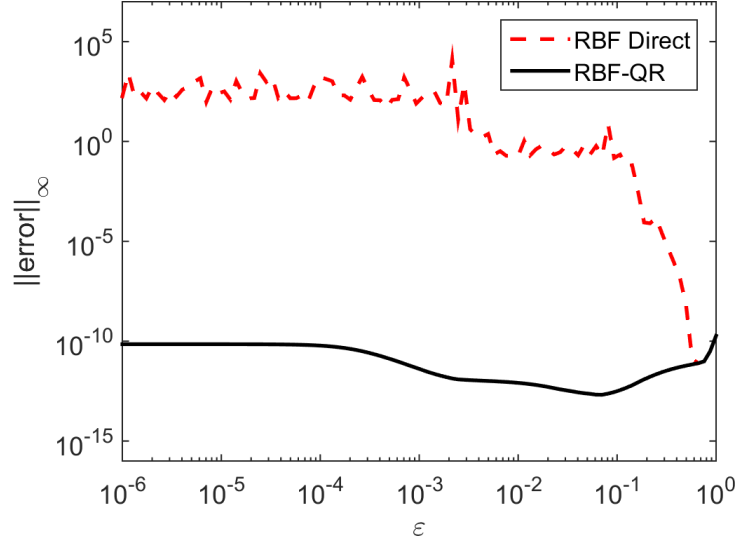


Figure 2.3: Log-log plot of the max norm error vs. values of ε for the target function $f = \sin(xyz)$. Here $n = 529$, and the IMQ RBF kernel was used. Note that for larger values of ε , the RBF Direct and RBF-QR methods give equivalent results. Though not clearly visible in the figure, this equivalence is demonstrated where the black line lies on top of the dashed red line.

problem together with the RBF Direct method. We see in this figure that around

$\varepsilon = 1$, the direct RBF interpolation method becomes numerically unstable, and the error quickly spikes. The errors from the Scalar RBF-QR approximations continue to decay beyond $\varepsilon = 1$ before increasing slightly in the flat limit. The increase in error as $\varepsilon \rightarrow 0$ corresponds to the basis functions converging to the spherical harmonics basis. We can infer from this that the Scalar RBF-QR algorithm can provide smaller errors than both the RBF direct and spherical harmonic interpolation processes for $\varepsilon \rightarrow 0$.

2.2.4 The Size of n

We conclude by commenting on the case when n is not a perfect square, which was not discussed with much detail in [7]. To illustrate the issues, consider the case when $n = (\mu_0 + 1)^2 - 2$, with $\mu_0 > 0$. The QR procedure proceeds almost entirely as described above, with the only change being in the E_1 and E_2 matrices, and thus the \tilde{E} matrix. E_1 and E_2 , for this case, are given by

$$E_1 = \begin{bmatrix} 1 & & & & \\ & \varepsilon^2 I_3 & & & \\ & & \ddots & & \\ & & & \varepsilon^{2\mu_0-2} I_{2\mu_0-1} & \\ & & & & \varepsilon^{2\mu_0} I_{2\mu_0+1-2} \end{bmatrix}, \quad (2.12)$$

$$E_2 = \begin{bmatrix} \varepsilon^{2\mu_0} I_2 & & & & \\ & \varepsilon^{2\mu_0+2} I_{2\mu_0+3} & & & \\ & & \ddots & & \\ & & & \varepsilon^{2q} I_{2q+1} \end{bmatrix}, \quad (2.13)$$

where E_1 is again of size n and E_2 is of size $(m-n)$ -by- $(m-n)$. A direct computation shows \tilde{E} takes the values

$$\tilde{E} = \left[\begin{array}{c|c|c|c|c} \varepsilon^{2\mu_0} J_{1,2} & \varepsilon^{2\mu_0+2} J_{1,2\mu_0+3} & \varepsilon^{2\mu_0+4} J_{1,2\mu_0+5} & \cdots & \cdots & \varepsilon^{2q} J_{1,2q+1} \\ \hline \varepsilon^{2\mu_0-2} J_{3,2} & \varepsilon^{2\mu_0} J_{3,2\mu_0+3} & \varepsilon^{2\mu_0+2} J_{3,2\mu_0+5} & \cdots & \cdots & \varepsilon^{2q-2} J_{3,2q+1} \\ \hline \varepsilon^{2\mu_0-4} J_{5,2} & \varepsilon^{2\mu_0-2} J_{5,2\mu_0+3} & \varepsilon^{2\mu_0} J_{5,2\mu_0+5} & \cdots & \cdots & \varepsilon^{2q-4} J_{5,2q+1} \\ \hline \vdots & \vdots & \ddots & \ddots & \vdots & \\ \hline \varepsilon^2 J_{2\mu_0-1,2} & \varepsilon^4 J_{2\mu_0-1,2\mu_0+3} & \varepsilon^6 J_{2\mu_0-1,2\mu_0+5} & \cdots & \cdots & \varepsilon^{2q-2\mu_0+2} J_{2\mu_0-1,2q+1} \\ \hline \varepsilon^0 J_{2\mu_0-1,2} & \varepsilon^2 J_{2\mu_0-1,2\mu_0+3} & \varepsilon^4 J_{2\mu_0-1,2\mu_0+5} & \cdots & \cdots & \varepsilon^{2q-2\mu_0} J_{2\mu_0-1,2q+1} \end{array} \right].$$

Note the block of all ones in the lower left corner. Thus, with this \tilde{E} in (2.2.2), each term in the new basis $\tilde{B}y$ consists of a spherical harmonic plus some perturbation, except for the last $2\mu_0 - 1$ terms which consist of an $O(1)$ linear combination of three spherical harmonics plus some small ε^2 perturbation. The specific additional spherical harmonics in these last $2\mu_0 - 1$ terms will not differ per radial kernel, but the weights in the linear combination will. Thus, in the $\varepsilon \rightarrow 0$ limit, the resulting RBF interpolant is not likely to be unique for different radial kernels.

CHAPTER 3

VECTOR RBF-QR ALGORITHM

The main result of this thesis is the Vector RBF-QR algorithm. This work synthesizes that of surface divergence-free RBF interpolation (Narcowich, Ward, & Wright [23]), surface curl-free RBF interpolation (Fuselier & Wright [13]), and the RBF-QR algorithm for scalar-valued functions on the sphere (Fornberg & Piret [7]). The Vector RBF-QR algorithm is an extension of the RBF-QR algorithm of Fornberg and Piret that allows for the stable computation of surface divergence-free and curl-free matrix-valued RBF interpolants in the flat limit. We derive this algorithm in similar fashion to the Scalar RBF-QR algorithm first for surface divergence-free RBF interpolants, with the process for surface curl-free RBF interpolants being a direct result.

3.1 Vector-Valued RBF Interpolation in the Flat Limit

Recall from (1.16) that Ψ_{div} is constructed from the scalar-valued radial kernel, ϕ , which is dependent on the shape parameter, ε . It is perhaps not surprising, then, that the conditioning of the linear system (1.23) and the accuracy of the interpolant (1.20) are dependent on the shape parameter in the same way as the scalar RBF interpolant (1.5), i.e., larger values of ε lead to a poor approximation of the target field while smaller values of ε provide better approximations of the target field. Also similar to the case for scalar-valued interpolants, ill-conditioning enters

the system (1.23) in the flat limit. As $\varepsilon \rightarrow 0$, all of the entries in the interpolation matrix become 0, causing the system to be singular. This relationship between ill-conditioning and accuracy of an interpolation problem of a divergence-free vector field on \mathbb{S}^2 when it is computed via (1.23) is illustrated in Figure 3.1. By extending the Scalar RBF-QR algorithm for use with surface matrix-valued kernels, we develop the first numerically stable algorithm for approximating divergence-free and curl-free vector fields on \mathbb{S}^2 in the flat limit.

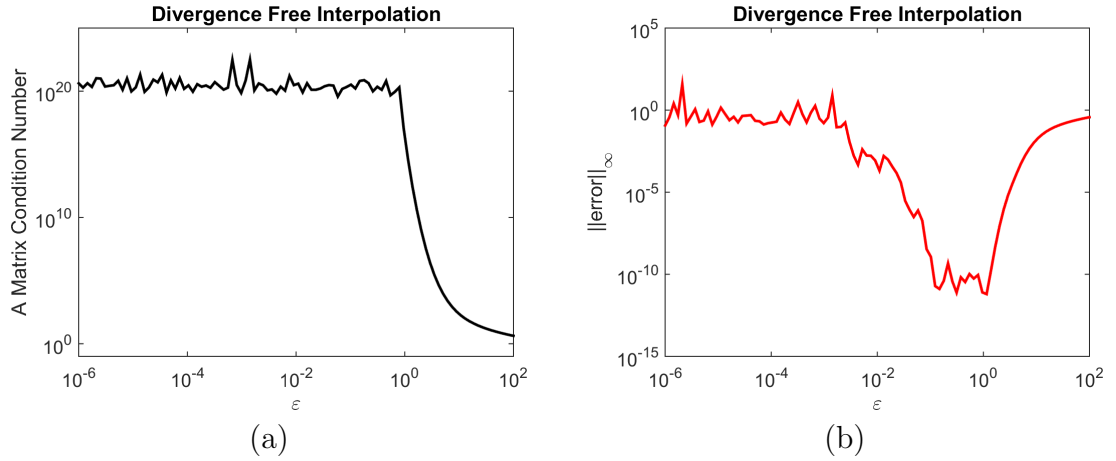


Figure 3.1: A problem illustrating the ill-conditioning that enters the interpolation process in the RBF Direct method for $n = 528$ quasi-uniformly distributed nodes and target function Ψ used in the second numerical test in Chapter 4. (a) Condition number of the $A_{\Psi_{div}}$ matrix from (1.23) vs. ε (b) Max norm error vs ε in the surface divergence-free RBF interpolant using the RBF Direct approach

3.2 Vector RBF-QR Algorithm for Surface Divergence-Free RBFs

3.2.1 Vector Spherical Harmonic Expansion

Similar to the Scalar RBF-QR algorithm of Fornberg and Piret [7], the key idea behind the Vector RBF-QR algorithm is to replace the ill-conditioned matrix-valued basis

with a better basis built from vector spherical harmonic expansions; see Section 1.4.2. These expansions arise naturally from the scalar spherical harmonic expansions of $\phi(\|\mathbf{x} - \mathbf{y}_j\|)$ given in (2.2). For example, the surface divergence-free kernel (1.16) can be expanded as follows:

$$\begin{aligned}\Psi_{div}(\mathbf{x}, \mathbf{y}_j) &= Q_{\mathbf{x}} \left(\nabla_{\mathbf{x}} \nabla_{\mathbf{y}}^T \phi(\|\mathbf{x} - \mathbf{y}_j\|) \right) Q_{\mathbf{y}} \\ &= \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \left\{ \varepsilon^{2\mu} c_{\mu,\varepsilon} Q_{\mathbf{x}} \nabla_{\mathbf{x}} Y_{\mu}^{\nu}(\mathbf{x}) \right\} \left(Q_{\mathbf{y}} \nabla_{\mathbf{y}} Y_{\mu}^{\nu}(\mathbf{y}_j) \right)^T \\ &= \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \left\{ \varepsilon^{2\mu} c_{\mu,\varepsilon} \mathbf{w}_{\mu}^{\nu}(\mathbf{x}) \right\} \left(\mathbf{w}_{\mu}^{\nu}(\mathbf{y}_j) \right)^T,\end{aligned}\tag{3.1}$$

where the expansion is now in terms of the non-normalized divergence-free vector spherical harmonics defined in (1.34) and $c_{\mu,\varepsilon}$ are as defined in Table 2.1.

As in the description of the Scalar RBF-QR algorithm, we will put a condition on the number of interpolation nodes, n , to simplify the presentation of the algorithm below. Since we removed the constant spherical harmonic function from the expansion, we let $n = (\mu_0 + 1)^2 - 1$, for some $\mu_0 > 0$, in order to ensure a unique way to split the matrices involved in the algorithm.

3.2.2 Matrix Representation and QR Factorization

Recall from Section 1.3.2 that in order to interpolate divergence-free vector fields tangent to the sphere with the matrix-valued divergence-free interpolant, we must represent the coefficient vectors and target field samples in terms of the orthonormal tangent basis vectors (1.19). In (1.23) we saw that this is equivalent to representing the kernel Ψ_{div} in terms of these basis vectors. We will denote this kernel as $\tilde{\Psi}_{div}$:

$$\tilde{\Psi}_{div}(\mathbf{x}, \mathbf{y}_j) = \begin{bmatrix} \mathbf{d}_{\mathbf{x}}^T \\ \mathbf{e}_{\mathbf{x}}^T \end{bmatrix} \Psi_{div}(\mathbf{x}, \mathbf{y}_j) \begin{bmatrix} \mathbf{d}_j & \mathbf{e}_j \end{bmatrix}.\tag{3.2}$$

Using (3.1) on the right-hand side of (3.2) gives the expansion

$$\tilde{\Psi}_{div}(\mathbf{x}, \mathbf{y}_j) = \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \left(\varepsilon^{2\mu} c_{\mu,\varepsilon} \begin{bmatrix} \mathbf{d}_{\mathbf{x}}^T \\ \mathbf{e}_{\mathbf{x}}^T \end{bmatrix} \mathbf{w}_{\mu}^{\nu}(\mathbf{x}) \right) (\mathbf{w}_{\mu}^{\nu}(\mathbf{y}_j))^T \begin{bmatrix} \mathbf{d}_j & \mathbf{e}_j \end{bmatrix}. \quad (3.3)$$

This is a 2-by-2 matrix whose entries are in terms of the meridional and zonal divergence-free vector spherical harmonics:

$$\tilde{\Psi}_{div}(\mathbf{x}, \mathbf{y}_j) = \left[\begin{array}{c|c} (a) & (b) \\ \hline (c) & (d) \end{array} \right], \text{ where} \quad (3.4)$$

$$\begin{aligned} (a) \quad & \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} \mathbf{d}_{\mathbf{x}}^T \mathbf{w}_{\mu}^{\nu}(\mathbf{x}) \} (\mathbf{w}_{\mu}^{\nu}(\mathbf{y}_j))^T \mathbf{d}_j = \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} G_{\mu}^{\nu}(\mathbf{x}) \} G_{\mu}^{\nu}(\mathbf{y}_j) \\ (b) \quad & \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} \mathbf{d}_{\mathbf{x}}^T \mathbf{w}_{\mu}^{\nu}(\mathbf{x}) \} (\mathbf{w}_{\mu}^{\nu}(\mathbf{y}_j))^T \mathbf{e}_j = \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} G_{\mu}^{\nu}(\mathbf{x}) \} H_{\mu}^{\nu}(\mathbf{y}_j) \\ (c) \quad & \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} \mathbf{e}_{\mathbf{x}}^T \mathbf{w}_{\mu}^{\nu}(\mathbf{x}) \} (\mathbf{w}_{\mu}^{\nu}(\mathbf{y}_j))^T \mathbf{d}_j = \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} H_{\mu}^{\nu}(\mathbf{x}) \} G_{\mu}^{\nu}(\mathbf{y}_j) \\ (d) \quad & \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} \mathbf{e}_{\mathbf{x}}^T \mathbf{w}_{\mu}^{\nu}(\mathbf{x}) \} (\mathbf{w}_{\mu}^{\nu}(\mathbf{y}_j))^T \mathbf{e}_j = \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} H_{\mu}^{\nu}(\mathbf{x}) \} H_{\mu}^{\nu}(\mathbf{y}_j). \end{aligned}$$

As in (2.4) of the Scalar RBF-QR algorithm, we want to represent the vector containing the tangent basis functions $\tilde{\Psi}_{div}(\mathbf{x}, \mathbf{y}_j)$, $j = 1, \dots, n$ as an infinite matrix product in terms of the divergence-free vector spherical harmonic expansions. Using (3.4), we see that this “vector” is a $2n$ -by-2 system of the form

$$\begin{bmatrix} \tilde{\Psi}_{div}(\mathbf{x}, \mathbf{y}_1) \\ \vdots \\ \tilde{\Psi}_{div}(\mathbf{x}, \mathbf{y}_n) \end{bmatrix} = \begin{bmatrix} \left[\begin{array}{c|c} \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} G_{\mu}^{\nu}(\mathbf{x}) \} G_{\mu}^{\nu}(\mathbf{y}_1) & \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} G_{\mu}^{\nu}(\mathbf{x}) \} H_{\mu}^{\nu}(\mathbf{y}_1) \\ \hline \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} H_{\mu}^{\nu}(\mathbf{x}) \} G_{\mu}^{\nu}(\mathbf{y}_1) & \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} H_{\mu}^{\nu}(\mathbf{x}) \} H_{\mu}^{\nu}(\mathbf{y}_1) \end{array} \right] \\ \vdots \\ \left[\begin{array}{c|c} \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} G_{\mu}^{\nu}(\mathbf{x}) \} G_{\mu}^{\nu}(\mathbf{y}_n) & \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} G_{\mu}^{\nu}(\mathbf{x}) \} H_{\mu}^{\nu}(\mathbf{y}_n) \\ \hline \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} H_{\mu}^{\nu}(\mathbf{x}) \} G_{\mu}^{\nu}(\mathbf{y}_n) & \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} H_{\mu}^{\nu}(\mathbf{x}) \} H_{\mu}^{\nu}(\mathbf{y}_n) \end{array} \right] \end{bmatrix}. \quad (3.5)$$

We can rewrite this as the following infinite block matrix-vector product,

$$\begin{aligned}
& \begin{bmatrix} \tilde{\Psi}_{div}(\mathbf{x}, \mathbf{y}_1) \\ \vdots \\ \tilde{\Psi}_{div}(\mathbf{x}, \mathbf{y}_n) \end{bmatrix} = \\
& = \begin{bmatrix} c_{\mu,\varepsilon} G_1^{-1}(\mathbf{y}_1) & \frac{c_{0,\varepsilon}}{2} G_1^0(\mathbf{y}_1) & c_{\mu,\varepsilon} G_1^1(\mathbf{y}_1) \cdots \\ c_{\mu,\varepsilon} H_1^{-1}(\mathbf{y}_1) & \frac{c_{0,\varepsilon}}{2} H_1^0(\mathbf{y}_1) & c_{\mu,\varepsilon} H_1^1(\mathbf{y}_1) \cdots \\ \vdots & \vdots & \vdots \\ c_{\mu,\varepsilon} G_1^{-1}(\mathbf{y}_N) & \frac{c_{0,\varepsilon}}{2} G_1^0(\mathbf{y}_N) & c_{\mu,\varepsilon} G_1^1(\mathbf{y}_N) \cdots \\ c_{\mu,\varepsilon} H_1^{-1}(\mathbf{y}_N) & \frac{c_{0,\varepsilon}}{2} H_1^0(\mathbf{y}_N) & c_{\mu,\varepsilon} H_1^1(\mathbf{y}_N) \cdots \end{bmatrix} \begin{bmatrix} \varepsilon^2 & & & \\ & \varepsilon^2 & & \\ & & \varepsilon^2 & \\ & & & \ddots & \ddots \end{bmatrix} \begin{bmatrix} G_1^{-1}(\mathbf{x}) & H_1^{-1}(\mathbf{x}) \\ G_1^0(\mathbf{x}) & H_1^0(\mathbf{x}) \\ G_1^1(\mathbf{x}) & H_1^1(\mathbf{x}) \\ \vdots & \vdots \end{bmatrix} \\
& = B^\infty E^\infty Y^\infty.
\end{aligned} \tag{3.6}$$

The first step of the Vector RBF-QR algorithm is to truncate these infinite matrices using a vector spherical harmonic degree value $\mu = k$. There are two stipulations for this truncation degree. First, k must be at least as big as the degree needed to ensure that there will be enough vector spherical harmonic terms in the expansion to approximate our basis to machine precision (μ_{trunc}). Additionally, k must be larger than $\sqrt{2}\mu_0 + 1$ so that we have proper dimensions for partitioning the matrices involved in the algorithm. In order to achieve this, we choose $k \geq \max[\mu_{trunc}, \lceil \sqrt{2}\mu_0 + 1 \rceil]$. We then denote the truncated matrix product from (3.5) as

$$\underbrace{\begin{bmatrix} \tilde{\Psi}_{div}(\mathbf{x}, \mathbf{y}_1) \\ \vdots \\ \tilde{\Psi}_{div}(\mathbf{x}, \mathbf{y}_n) \end{bmatrix}}_{\tilde{P}_{div}} \approx \begin{bmatrix} & B & \\ & & \end{bmatrix} \begin{bmatrix} & E & \\ & & \end{bmatrix} \begin{bmatrix} Y \\ \\ \end{bmatrix}. \tag{3.7}$$

By letting $m = (k + 1)^2 - 1$, we have that the size of B is $2n$ -by- m .

Before describing the second step of the Vector RBF-QR algorithm, we establish dimensions and structures of the matrices involved. The system (3.7) is of the form

$$\underbrace{\begin{bmatrix} B_1 & | & B_2 & | & \cdots & | & B_{\mu_0} & | & B_{\mu_0+1} & | & \cdots & | & B_k \end{bmatrix}}_B EY,$$

where B_μ , $1 \leq \mu \leq k$, are the block matrices of size $2n$ -by- $(2\mu + 1)$ with block entries

$$(B_\mu)_{i,j} = \begin{cases} \begin{bmatrix} c_{\mu,\varepsilon} G_\mu^{j-(\mu+1)}(\mathbf{y}_i) \\ c_{\mu,\varepsilon} H_\mu^{j-(\mu+1)}(\mathbf{y}_i) \end{bmatrix} & j \neq \mu + 1, \\ \begin{bmatrix} \frac{c_{\mu,\varepsilon}}{2} G_\mu^0(\mathbf{y}_i) \\ \frac{c_{\mu,\varepsilon}}{2} H_\mu^0(\mathbf{y}_i) \end{bmatrix} & j = \mu + 1, \end{cases} \quad j = 1, \dots, 2\mu + 1, \quad i = 1, \dots, n.$$

The diagonal E matrix can be written as two square, diagonal blocks, E_1 and E_2

$$E = \left[\begin{array}{c|c} E_1 & \\ \hline & E_2 \end{array} \right],$$

where

$$E_1 = \begin{bmatrix} \varepsilon^2 I_3 & & & \\ & \varepsilon^4 I_5 & & \\ & & \ddots & \\ & & & \varepsilon^{2\mu_0} I_{2\mu_0+1} \end{bmatrix}, \quad (3.8)$$

$$E_2 = \begin{bmatrix} \varepsilon^{2\mu_0+2} I_{2\mu_0+3} & & & \\ & \varepsilon^{2\mu_0+4} I_{2\mu_0+5} & & \\ & & \ddots & \\ & & & \varepsilon^{2k} I_{2k+1} \end{bmatrix}, \quad (3.9)$$

and I_μ is the identity matrix of size μ -by- μ . Due to the truncation and restriction on n , we see that E_1 is of size n -by- n , and E_2 is of size $(m - n)$ -by- $(m - n)$. Finally, the Y vector is given by

$$Y = \begin{bmatrix} \frac{Y_1}{Y_2} \\ \vdots \\ \frac{Y_{\mu_0}}{Y_{\mu_0+1}} \\ \vdots \\ \frac{Y_k}{Y_k} \end{bmatrix},$$

$$(Y_\mu)_{j,1} = G_\mu^{j-(\mu+1)}(\mathbf{x}), \quad (Y_\mu)_{j,2} = H_\mu^{j-(\mu+1)}(\mathbf{x}), \quad j = 1, \dots, 2\mu + 1.$$

In the flat limit, our new basis is still highly ill-conditioned because of the powers of ε in (3.6). However, all of these powers of ε are confined to the E matrix. Recall that the $c_{\mu,\varepsilon}$ do not affect the conditioning of the system. In order to develop a better conditioned basis, we move to the next step of the algorithm.

The second step of the Vector RBF-QR algorithm is to perform a QR factorization on B . Recall that the QR factorization only operates on the columns of a matrix without combining terms in successive columns. Computing a QR factorization of B gives

$$\tilde{P}_{div} \approx Q \underbrace{[R_1 | R_2]}_R \left[\frac{E_1}{E_2} \right] Y, \quad (3.10)$$

With the goal of analytically removing the issues with small ε , we partition R into R_1 and R_2 , where R_1 is $2n$ -by- $2n$ upper-triangular, and R_2 is an $2n$ -by- $(m - 2n)$ full matrix. Notice that R_1 is $2n$ -by- $2n$ as opposed to size n -by- n in the Scalar RBF-QR algorithm since we are interpolating a vector field of n points, each with 2 components.

The third step of the Vector RBF-QR algorithm is to perform a clever factoring of the expression on the right-hand side of (3.10). We assume that the diagonal entries

of R_1 are non-zero so that it is invertible¹, which allows us to re-write the system on the right-hand side of (3.10) as

$$\tilde{P}_{div} \approx QR_1 [I_{2n} | R_1^{-1}R_2] \left[\frac{E_1}{E_2} \right] Y.$$

The diagonal structure of E allow us to again re-write this expression as

$$\begin{aligned} \tilde{P}_{div} &\approx QR_1 [E_1 | R_1^{-1}R_2E_2] Y \\ &= QR_1 E_1 \underbrace{[I_{2n} | E_1^{-1}R_1^{-1}R_2E_2]}_{\tilde{B}_{div}} Y. \end{aligned} \quad (3.11)$$

It follows from this new expression that any element in the span of \tilde{P}_{div} can be represented to machine precision by a linear combination of the elements of $\tilde{B}_{div}Y$.

The fourth and final step of the Vector RBF-QR algorithm is to reformulate \tilde{B}_{div} using properties of the Hadamard product of diagonal matrices. We will also use this to show that this new basis is much better conditioned than the original in the flat limit. To begin the final step, we consider the product in the second block-column of \tilde{B}_{div} from (3.11). Using the properties of multiplication of a matrix on the left and right by diagonal matrices (see Appendix A), we have

$$E_1^{-1}R_1^{-1}R_2E_2 = (R_1^{-1}R_2) \circ \underbrace{(E_1^{-1}J_{2n,m-2n}E_2)}_{\tilde{E}},$$

where $J_{2n,m-2n}$ is a matrix with all entries being 1 and \circ denotes the Hadamard product, or entry-wise multiplication. After considering the structure of E_1 and E_2 given in (3.8) and (3.9), respectively, we see that the entries of \tilde{E} are given explicitly by

¹This will be true if the nodes are unisolvent with respect to the divergence-free vector spherical harmonic basis.

$$\left[\begin{array}{c|c|c|c} \varepsilon^{2\mu_0} J_{3,2\mu_0+3} & \varepsilon^{2\mu_0+2} J_{3,2\mu_0+5} & \cdots & \cdots & \varepsilon^{2k-2} J_{3,2k+1} \\ \hline \varepsilon^{2\mu_0-2} J_{5,2\mu_0+3} & \varepsilon^{2\mu_0} J_{5,2\mu_0+5} & \cdots & \cdots & \varepsilon^{2k-4} J_{5,2k+1} \\ \hline \vdots & \vdots & \ddots & \ddots & \vdots \\ \hline \varepsilon^4 J_{2\mu_0-1,2\mu_0+3} & \varepsilon^6 J_{2\mu_0-1,2\mu_0+5} & \cdots & \cdots & \varepsilon^{2k-2\mu_0+2} J_{2\mu_0-1,2k+1} \\ \hline \varepsilon^2 J_{2\mu_0+1,2\mu_0+3} & \varepsilon^4 J_{2\mu_0+1,2\mu_0+5} & \cdots & \cdots & \varepsilon^{2k-2\mu_0} J_{2\mu_0+1,2k+1} \end{array} \right]. \quad (3.12)$$

Rewriting $\tilde{B}_{div}Y$ as

$$\tilde{B}_{div}Y = \left[I_{2n} \mid (R_1^{-1}R_2) \circ \tilde{E} \right] Y,$$

we see that the rows of $\tilde{B}_{div}Y$ are a better basis for $\text{span} \{ \tilde{\Psi}_{div}(\cdot, \mathbf{y}_j) \}_{j=1}^n$. Thus, we have found a basis where the ill-conditioning associated with small ε has been analytically removed. Note that the first $2k$ terms of \tilde{B}_{div} are the vector spherical harmonic basis functions and all subsequent terms are $\mathcal{O}(\varepsilon^2)$. So the matrix-valued divergence-free RBF interpolant converges to a divergence-free vector spherical harmonic interpolant in the flat limit.

Note that just as with the Scalar RBF-QR algorithm, it is possible to include the spherical harmonic coefficients $c_{\mu,\varepsilon}$ in the diagonal matrices E_1 and E_2 and generate a similar analytical simplification for \tilde{E} . This has the added advantage of removing all ε dependence in the actual QR numerical computation, and thus removes any numerical contamination for small ε . This approach unifies the Vector RBF-QR method for all radial kernels since, for the same set of nodes, the only thing that would change is the \tilde{E} matrix.

3.3 Vector RBF-QR Algorithm for Surface Curl-Free RBFs

We now briefly discuss the process for surface curl-free RBF kernels (1.26), noting only the differences in notation. Just as before, we can expand the kernel Ψ_{curl} utilizing the tangential curl-free vector spherical harmonics:

$$\begin{aligned}
\Psi_{curl}(\mathbf{x}, \mathbf{y}_j) &= P_{\mathbf{x}} \left(\nabla_{\mathbf{x}} \nabla_{\mathbf{y}}^T \phi(|\mathbf{x} - \mathbf{y}_j|) \right) P_{\mathbf{y}} \\
&= \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} P_{\mathbf{x}} \nabla_{\mathbf{x}} Y_{\mu}^{\nu}(\mathbf{x}) \} \left(P_{\mathbf{y}} \nabla_{\mathbf{y}} Y_{\mu}^{\nu}(\mathbf{y}_j) \right)^T \\
&= \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} \mathbf{z}_{\mu}^{\nu}(\mathbf{x}) \} \left(\mathbf{z}_{\mu}^{\nu}(\mathbf{y}_j) \right)^T,
\end{aligned} \tag{3.13}$$

where the expansion is now in terms of the non-normalized curl-free vector spherical harmonics defined in (1.35) and $c_{\mu,\varepsilon}$ are as defined in Table 2.1. We impose the same restriction on n as before and represent the kernel Ψ_{curl} in terms of the orthonormal tangent basis vectors (1.19). We will denote this kernel as $\tilde{\Psi}_{curl}$:

$$\tilde{\Psi}_{curl}(\mathbf{x}, \mathbf{y}_j) = \begin{bmatrix} \mathbf{d}_{\mathbf{x}}^T \\ \mathbf{e}_{\mathbf{x}}^T \end{bmatrix} \Psi_{curl}(\mathbf{x}, \mathbf{y}_j) \begin{bmatrix} \mathbf{d}_j & \mathbf{e}_j \end{bmatrix}. \tag{3.14}$$

Using (3.13) on the right-hand side of (3.14) gives the expansion

$$\tilde{\Psi}_{curl}(\mathbf{x}, \mathbf{y}_j) = \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \left(\varepsilon^{2\mu} c_{\mu,\varepsilon} \begin{bmatrix} \mathbf{d}_{\mathbf{x}}^T \\ \mathbf{e}_{\mathbf{x}}^T \end{bmatrix} \mathbf{z}_{\mu}^{\nu}(\mathbf{x}) \right) \left(\mathbf{z}_{\mu}^{\nu}(\mathbf{y}_j) \right)^T \begin{bmatrix} \mathbf{d}_j & \mathbf{e}_j \end{bmatrix}. \tag{3.15}$$

This is a 2-by-2 matrix whose entries are in terms of the meridional and zonal curl-free vector spherical harmonics:

$$\tilde{\Psi}_{curl}(\mathbf{x}, \mathbf{y}_j) = \left[\begin{array}{c|c} (a) & (b) \\ \hline (c) & (d) \end{array} \right], \text{ where} \tag{3.16}$$

$$\begin{aligned}
(a) \quad & \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} \mathbf{d}_{\mathbf{x}}^T \mathbf{z}_{\mu}^{\nu}(\mathbf{x}) \} \left(\mathbf{z}_{\mu}^{\nu}(\mathbf{y}_j) \right)^T \mathbf{d}_j = \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} K_{\mu}^{\nu}(\mathbf{x}) \} K_{\mu}^{\nu}(\mathbf{y}_j) \\
(b) \quad & \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} \mathbf{d}_{\mathbf{x}}^T \mathbf{z}_{\mu}^{\nu}(\mathbf{x}) \} \left(\mathbf{z}_{\mu}^{\nu}(\mathbf{y}_j) \right)^T \mathbf{e}_j = \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} K_{\mu}^{\nu}(\mathbf{x}) \} L_{\mu}^{\nu}(\mathbf{y}_j) \\
(c) \quad & \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} \mathbf{e}_{\mathbf{x}}^T \mathbf{z}_{\mu}^{\nu}(\mathbf{x}) \} \left(\mathbf{z}_{\mu}^{\nu}(\mathbf{y}_j) \right)^T \mathbf{d}_j = \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} L_{\mu}^{\nu}(\mathbf{x}) \} K_{\mu}^{\nu}(\mathbf{y}_j) \\
(d) \quad & \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} \mathbf{e}_{\mathbf{x}}^T \mathbf{z}_{\mu}^{\nu}(\mathbf{x}) \} \left(\mathbf{z}_{\mu}^{\nu}(\mathbf{y}_j) \right)^T \mathbf{e}_j = \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} L_{\mu}^{\nu}(\mathbf{x}) \} L_{\mu}^{\nu}(\mathbf{y}_j).
\end{aligned}$$

As in (2.4) of the Scalar RBF-QR algorithm, we want to represent our vector of

tangent basis functions $\tilde{\Psi}_{curl}(\mathbf{x}, \mathbf{y}_j)$, $j = 1, \dots, n$ as an infinite matrix product in terms of the curl-free vector spherical harmonic expansions. Using (3.16), we see that this is a $2n$ -by- 2 system of the form:

$$\begin{bmatrix} \tilde{\Psi}_{curl}(\mathbf{x}, \mathbf{y}_1) \\ \vdots \\ \tilde{\Psi}_{curl}(\mathbf{x}, \mathbf{y}_n) \end{bmatrix} = \begin{bmatrix} \left[\begin{array}{c|c} \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} K_{\mu}^{\nu}(\mathbf{x}) \} K_{\mu}^{\nu}(\mathbf{y}_1) & \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} K_{\mu}^{\nu}(\mathbf{x}) \} L_{\mu}^{\nu}(\mathbf{y}_1) \\ \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} L_{\mu}^{\nu}(\mathbf{x}) \} K_{\mu}^{\nu}(\mathbf{y}_1) & \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} L_{\mu}^{\nu}(\mathbf{x}) \} L_{\mu}^{\nu}(\mathbf{y}_1) \end{array} \right] \\ \vdots \\ \left[\begin{array}{c|c} \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} K_{\mu}^{\nu}(\mathbf{x}) \} K_{\mu}^{\nu}(\mathbf{y}_n) & \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} K_{\mu}^{\nu}(\mathbf{x}) \} L_{\mu}^{\nu}(\mathbf{y}_n) \\ \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} L_{\mu}^{\nu}(\mathbf{x}) \} K_{\mu}^{\nu}(\mathbf{y}_n) & \sum_{\mu=1}^{\infty} \sum_{\nu=-\mu}^{\mu} \{ \varepsilon^{2\mu} c_{\mu,\varepsilon} L_{\mu}^{\nu}(\mathbf{x}) \} L_{\mu}^{\nu}(\mathbf{y}_n) \end{array} \right] \end{bmatrix}. \quad (3.17)$$

This then gives us the infinite block matrix-vector product,

$$\begin{bmatrix} \tilde{\Psi}_{curl}(\mathbf{x}, \mathbf{y}_1) \\ \vdots \\ \tilde{\Psi}_{curl}(\mathbf{x}, \mathbf{y}_n) \end{bmatrix} = \begin{bmatrix} c_{\mu,\varepsilon} K_1^{-1}(\mathbf{y}_1) & \frac{c_{0,\varepsilon}}{2} K_1^0(\mathbf{y}_1) & c_{\mu,\varepsilon} K_1^1(\mathbf{y}_1) \cdots \\ c_{\mu,\varepsilon} L_1^{-1}(\mathbf{y}_1) & \frac{c_{0,\varepsilon}}{2} L_1^0(\mathbf{y}_1) & c_{\mu,\varepsilon} L_1^1(\mathbf{y}_1) \cdots \\ \vdots & \vdots & \vdots \\ c_{\mu,\varepsilon} K_1^{-1}(\mathbf{y}_N) & \frac{c_{0,\varepsilon}}{2} K_1^0(\mathbf{y}_N) & c_{\mu,\varepsilon} K_1^1(\mathbf{y}_N) \cdots \\ c_{\mu,\varepsilon} L_1^{-1}(\mathbf{y}_N) & \frac{c_{0,\varepsilon}}{2} L_1^0(\mathbf{y}_N) & c_{\mu,\varepsilon} L_1^1(\mathbf{y}_N) \cdots \end{bmatrix} \begin{bmatrix} \varepsilon^2 & & & \\ & \varepsilon^2 & & \\ & & \ddots & \\ & & & \ddots \end{bmatrix} \begin{bmatrix} K_1^{-1}(\mathbf{x}) & L_1^{-1}(\mathbf{x}) \\ K_1^0(\mathbf{x}) & L_1^0(\mathbf{x}) \\ K_1^1(\mathbf{x}) & L_1^1(\mathbf{x}) \\ \vdots & \vdots \end{bmatrix}. \quad (3.18)$$

The system (3.18) is now of the same form as (3.6). Therefore, one can perform the steps of the Vector RBF-QR algorithm in the same manner as for the divergence-free case.

In the next chapter, we will provide numerical results for the Vector RBF-QR algorithm for both surface divergence-free and surface curl-free RBF interpolation.

3.4 Vector RBF-QR Algorithm for the Helmholtz-Hodge Decomposition of Surface Vector Fields

We conclude this chapter by providing the setup for the Vector RBF-QR algorithm for the Helmholtz-Hodge decomposition of surface vector fields. We recall from Section 1.3.2 the Fuselier and Wright found that the vector-valued RBF interpolant for any given tangent vector field on \mathbb{S}^2 is of the form (1.30) [13]. This idea of decomposing a tangent vector field interpolant into divergence-free and curl-free parts extends to the Vector RBF-QR algorithm. Using (1.30), we can write the kernel, Ψ , in terms of the tangent basis vectors (1.19) and thus in terms of $\tilde{\Psi}_{div}$ and $\tilde{\Psi}_{curl}$:

$$\begin{aligned}
\tilde{\Psi}(\mathbf{x}_i, \mathbf{y}_j) &= \begin{bmatrix} \mathbf{d}_i^T \\ \mathbf{e}_i^T \end{bmatrix} \Psi(\mathbf{x}_i, \mathbf{y}_j) [\mathbf{d}_j \ \mathbf{e}_j] \\
&= \begin{bmatrix} \mathbf{d}_i^T \\ \mathbf{e}_i^T \end{bmatrix} [\Psi_{div}(\mathbf{x}_i, \mathbf{y}_j) + \Psi_{curl}(\mathbf{x}_i, \mathbf{y}_j)] [\mathbf{d}_j \ \mathbf{e}_j] \\
&= \begin{bmatrix} \mathbf{d}_i^T \\ \mathbf{e}_i^T \end{bmatrix} \Psi_{div}(\mathbf{x}_i, \mathbf{y}_j) [\mathbf{d}_j \ \mathbf{e}_j] + \begin{bmatrix} \mathbf{d}_i^T \\ \mathbf{e}_i^T \end{bmatrix} \Psi_{curl}(\mathbf{x}_i, \mathbf{y}_j) [\mathbf{d}_j \ \mathbf{e}_j] \\
&= \tilde{\Psi}_{div}(\mathbf{x}_i, \mathbf{y}_j) + \tilde{\Psi}_{curl}(\mathbf{x}_i, \mathbf{y}_j).
\end{aligned} \tag{3.19}$$

We can then decompose the vector of tangent basis functions $\tilde{\Psi}(\mathbf{x}_i, \mathbf{y}_j)$, $j = 1, \dots, n$ accordingly,

$$\begin{bmatrix} \tilde{\Psi}(\mathbf{x}_i, \mathbf{y}_1) \\ \vdots \\ \tilde{\Psi}(\mathbf{x}_i, \mathbf{y}_n) \end{bmatrix} = \begin{bmatrix} \tilde{\Psi}_{div}(\mathbf{x}_i, \mathbf{y}_1) \\ \vdots \\ \tilde{\Psi}_{div}(\mathbf{x}_i, \mathbf{y}_n) \end{bmatrix} + \begin{bmatrix} \tilde{\Psi}_{curl}(\mathbf{x}_i, \mathbf{y}_1) \\ \vdots \\ \tilde{\Psi}_{curl}(\mathbf{x}_i, \mathbf{y}_n) \end{bmatrix}. \tag{3.20}$$

With this, we can follow the steps detailed in Sections 3.2 and 3.3 in order to implement the Vector RBF-QR algorithm. The next chapter will provide numerical results from the Vector RBF-QR algorithm for divergence-free tangent vector fields, curl-free tangent vector fields, and tangent vector fields that are neither divergence-free nor curl-free.

CHAPTER 4

NUMERICAL RESULTS FROM THE VECTOR RBF-QR ALGORITHM

In this chapter we report on various numerical tests that we performed with the new Vector RBF-QR algorithm for both surface divergence-free and surface curl-free vector field interpolation. The tests include vector fields of varying complexity, from a sum of vector spherical harmonics, to divergence-free and curl-free fields we generate from the respective surface curl and surface gradient of a sum of Gaussian bumps on the sphere. For each test, we report the results for both the MQ and IMQ kernels as listed in Table 1.1. We used minimum energy node sets in all tests as the sample points Y [32]. The specific node sets we used, consisting of $n = 120$ and $n = 528$ nodes, are illustrated in Figure 4.1. We first present the surface divergence-free results, then follow this with the surface curl-free results.

4.1 Surface Divergence-Free Vector Fields

For the first numerical test, we used the $n = 120$ minimum energy node set, pictured in Figure 4.1(a). The test field uses the non-normalized divergence-free vector spherical harmonics introduced in Chapter 1:

$$\mathbf{u} = Q_{\mathbf{x}} \nabla_{\mathbf{x}} \Psi(\mathbf{x}), \quad \text{where } \Psi(\mathbf{x}) = -\frac{1}{\sqrt{3}} Y_1^0(\mathbf{x}) + \frac{8}{3} \sqrt{\frac{2}{385}} Y_5^4(\mathbf{x}).$$

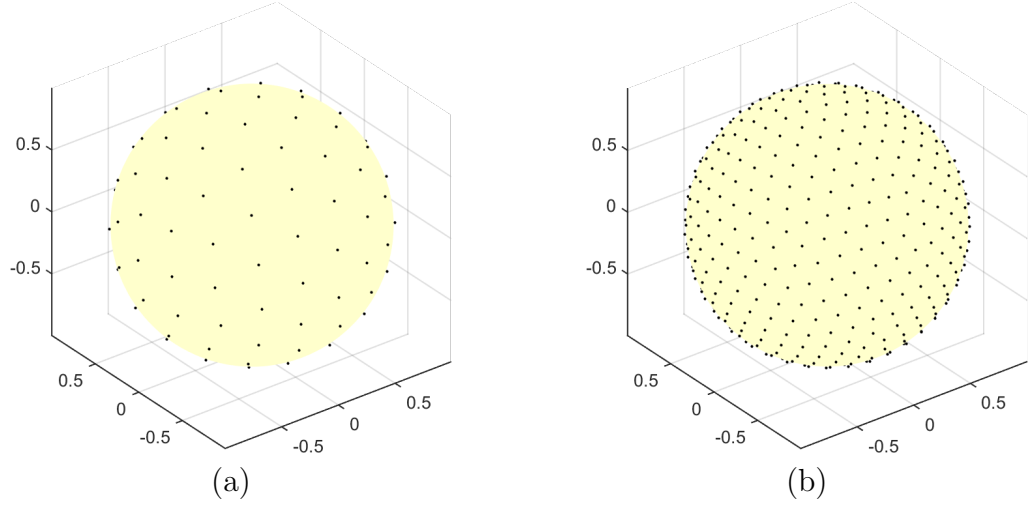


Figure 4.1: Minimum energy node sets used in the numerical experiments: (a) 120 nodes and (b) 528 nodes

Figure 4.2 illustrates this surface divergence-free field. In Figure 4.3, we present log-log plots of the max-norm error in the approximation of the true field against ε for the RBF Direct method and Vector RBF-QR method using the (a) MQ and (b) IMQ kernels. In this figure, we see that the RBF Direct method becomes

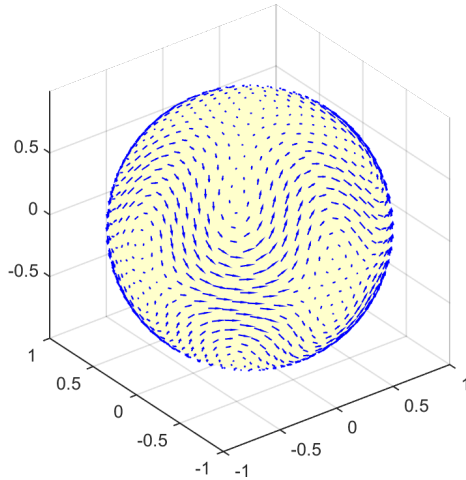


Figure 4.2: The surface divergence-free vector field to be interpolated for test 1.

numerically unstable around $\varepsilon = 0.1$, and the max-norm error increases in the flat

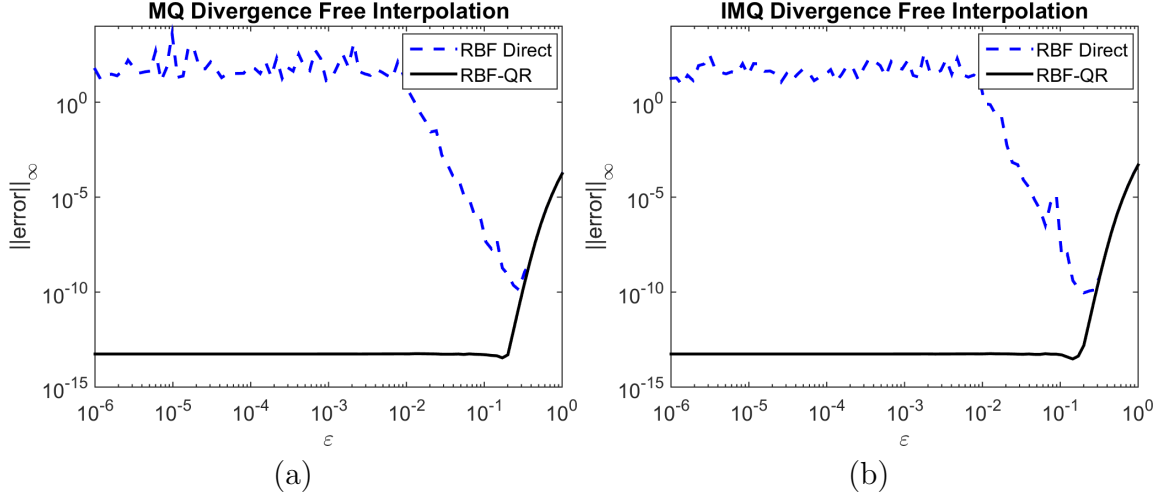


Figure 4.3: Numerical test 1: Log-log plot of the max-norm error in the approximation of the true field vs values of ε for both the RBF Direct method and the Vector RBF-QR method with the (a) MQ and (b) IMQ kernels.

limit as a result. The RBF-QR method, however, eliminates this ill-conditioning in the system. We see in Figure 4.3 that for both the MQ and IMQ kernels, the RBF-QR method is stable as $\varepsilon \rightarrow 0$ and achieves almost four more orders of accuracy than the RBF Direct method. We note that we recover the surface divergence-free vector spherical harmonic function to machine precision. This is to be expected since in the flat limit, the vector RBF interpolant converges to a vector spherical harmonic interpolant. Additionally, the node set used here is unisolvent with respect to the vector spherical harmonics, so the vector spherical harmonic interpolant is unique.

The second numerical test used the $n = 528$ minimum energy node set, pictured in Figure 4.1(b). In order to ensure that the vector field to be interpolated was divergence-free, we again took the surface-curl of a scalar-valued function:

$$\mathbf{u} = Q_{\mathbf{x}} \nabla_{\mathbf{x}} \Psi(\mathbf{x}), \text{ where}$$

$$\begin{aligned}
\Psi(\mathbf{x}) = & \exp\left(-0.01\left[\left(x - \frac{1}{\sqrt{3}}\right)^2 + \left(y - \frac{1}{\sqrt{3}}\right)^2 + \left(z - \frac{1}{\sqrt{3}}\right)^2\right]\right) \\
& + \exp\left(-0.01\left[\left(x + \frac{1}{\sqrt{3}}\right)^2 + \left(y - \frac{1}{\sqrt{3}}\right)^2 + \left(z - \frac{1}{\sqrt{3}}\right)^2\right]\right) \\
& + \exp\left(-0.01\left[\left(x + \frac{1}{\sqrt{3}}\right)^2 + \left(y + \frac{1}{\sqrt{3}}\right)^2 + \left(z - \frac{1}{\sqrt{3}}\right)^2\right]\right) \\
& + \exp\left(-0.01\left[\left(x - \frac{1}{\sqrt{3}}\right)^2 + \left(y + \frac{1}{\sqrt{3}}\right)^2 + \left(z + \frac{1}{\sqrt{3}}\right)^2\right]\right) \\
& + \exp\left(-0.01(z-1)^2\right) + \exp\left(-0.01(z+1)^2\right).
\end{aligned}$$

Figure 4.4 illustrates this surface divergence-free field. In Figure 4.5, we present log-log plots of the max-norm error in the approximation of the true field against ε for the RBF Direct method and Vector RBF-QR method using the (a) MQ and (b) IMQ kernels. We see in this figure that ill-conditioning enters the RBF Direct

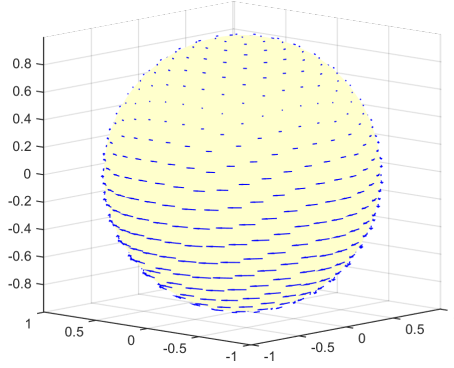


Figure 4.4: The surface divergence-free vector field to be interpolated in test 2.

method around $\varepsilon = 1$ and that the error increases rapidly as $\varepsilon \rightarrow 0$. We note here that in addition to remaining numerically stable in the flat limit, the RBF-QR method provides the smallest errors around $\varepsilon = 1$. Since we showed in the previous numerical test that the vector RBF interpolant converges to a vector spherical harmonic interpolant in the flat limit, we can conclude that the RBF-QR method achieves the best approximation of this target vector field for a value of ε that is untouchable for RBF Direct.

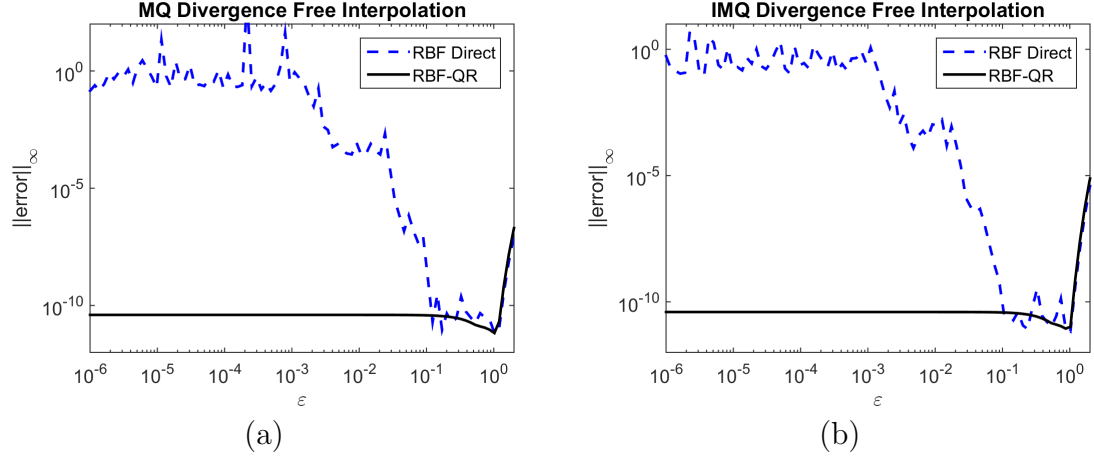


Figure 4.5: Numerical test 2: Log-log plot of the max-norm error in the approximation of the true field vs values of ε for both the RBF Direct method and the Vector RBF-QR method with the (a) MQ and (b) IMQ kernels.

For the third numerical test, we again used $n = 528$ minimum energy nodes. The field we interpolate here was first used by Narcowich, Ward, Fuselier, and Wright as an example of a non-smooth surface divergence-free vector field [12]. In order to describe the field, we begin by defining the function $g(t)$ as

$$g(t) = (2 - 2t)^{\frac{3}{2}}.$$

Next we let x and x_c be points on the unit sphere with respective spherical coordinates (θ, λ) and (θ_c, λ_c) . Then we define the dot product of these points as η , where

$$\eta = x^T x_c = \cos(\theta) \cos(\lambda - \lambda_c) \cos(\theta_c) + \sin(\theta) \sin(\theta_c).$$

We let $\eta_{\theta_c, \lambda_c}$ denote this dot product at the point (θ_c, λ_c) and define the field as

$$\mathbf{u} = Q_{\mathbf{x}} \nabla_{\mathbf{x}} \Psi(\mathbf{x}), \text{ where}$$

$$\Psi(\mathbf{x}) = g(\eta_{0, -\pi}) - g\left(\eta_{\frac{1}{10}, -\frac{\pi}{2}}\right) + 0.7g\left(\eta_{-\frac{\pi}{8}, 0}\right) - g\left(\eta_{-\frac{1}{10}, \frac{\pi}{2}}\right) + 0.3g\left(\eta_{\frac{\pi}{2} - \frac{1}{10}, 0}\right).$$

Figure 4.6 illustrates this field. The error plots in Figure 4.7 show that while the RBF-QR method with a non-zero ε achieves better accuracy than the vector spherical

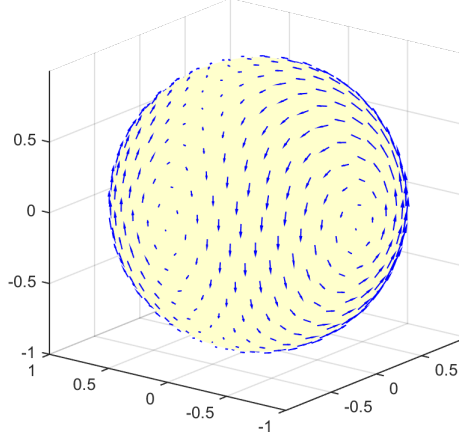


Figure 4.6: The surface divergence-free vector field to be interpolated for test 3.

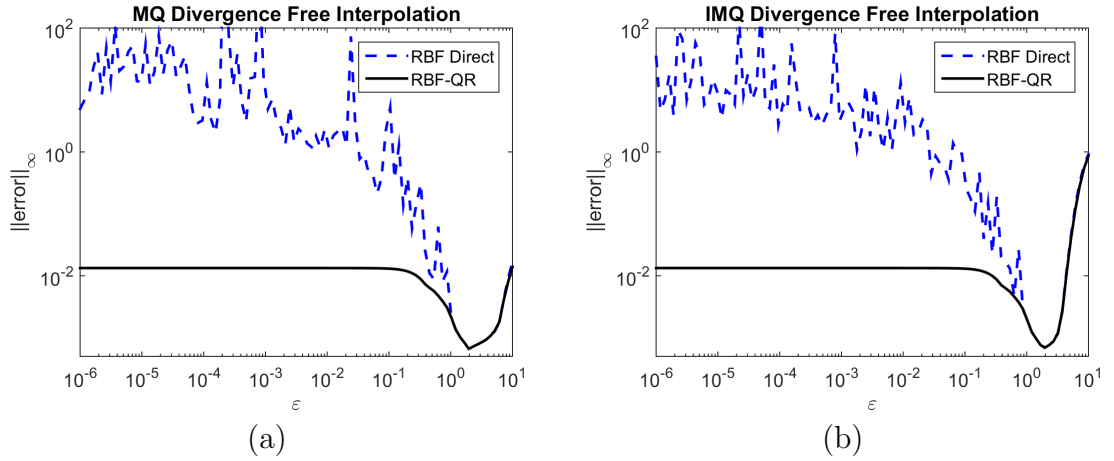


Figure 4.7: Numerical test 3: Log-log plot of the max-norm error in the approximation of the true field vs values of ε for both the RBF Direct method and the Vector RBF-QR method with the (a) MQ and (b) IMQ kernels.

harmonic interpolant, it does not provide better errors than those that can be obtained just using the RBF Direct method. This is due the roughness of the field used in this test. We conclude that while the Vector RBF-QR method performs well when interpolating smooth vector fields, it may not be preferable over the RBF Direct method when the fields are rough.

4.2 Surface Curl-Free Vector Fields

The first numerical test of the curl-free case used $n = 120$ minimum energy nodes. The test field used the non-normalized curl-free vector spherical harmonics introduced in Chapter 1:

$$\mathbf{u} = P_{\mathbf{x}} \nabla_{\mathbf{x}} \Psi(\mathbf{x}), \quad \text{where } \Psi(\mathbf{x}) = Y_4^0(\mathbf{x}) + Y_6^{-3}(\mathbf{x}).$$

Figure 4.8 illustrates this surface curl-free field. In Figure 4.9, we present log-log

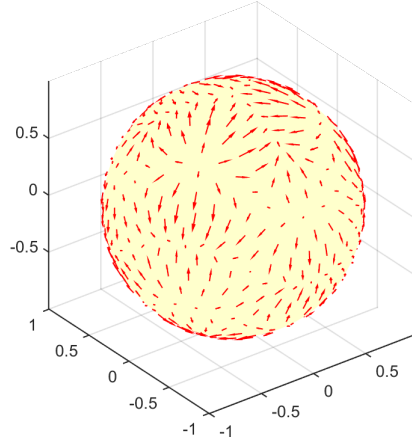


Figure 4.8: The surface curl-free vector field to be interpolated for test 1.

plots of the max-norm error in the approximation of the true field against ε for the RBF Direct method and Vector RBF-QR method using the (a) MQ and (b) IMQ kernels. Similar to the results from the first numerical test in the divergence-free case, we see that the RBF-QR method achieves four orders of accuracy over the RBF Direct method. We also note that we recover the surface curl-free vector spherical harmonic function to machine precision. This is again what we would expect due to the vector RBF interpolant converging to a vector spherical harmonic interpolant in the flat limit. Additionally, the vector spherical harmonic interpolant is unique since the node set used here is unisolvent with respect to the vector spherical harmonics.

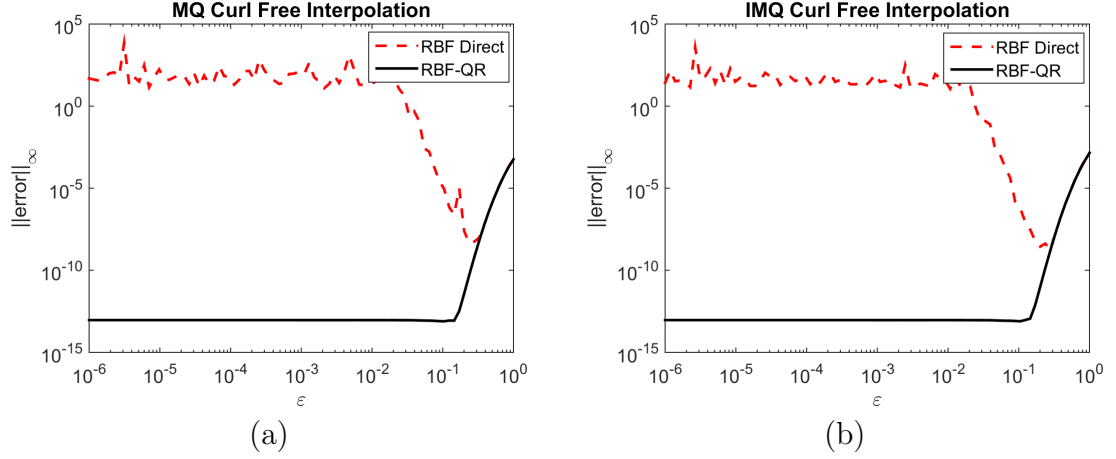


Figure 4.9: Curl-free numerical test 1: Log-log plot of the max-norm error in the approximation of the true field vs values of ε for both the RBF Direct method and the Vector RBF-QR method with the (a) MQ and (b) IMQ kernels.

The second numerical test of the curl-free case used the $n = 528$ minimum energy node set. In order to ensure that the vector field to be interpolated was curl-free, we again took the gradient of a scalar-valued function:

$$\mathbf{u} = P_{\mathbf{x}} \nabla_{\mathbf{x}} \Psi(\mathbf{x}), \quad \text{where}$$

$$\begin{aligned} \Psi(\mathbf{x}) = & \exp \left(-10 \left[\left(x - \frac{1}{\sqrt{3}} \right)^2 + \left(y - \frac{1}{\sqrt{3}} \right)^2 + \left(z - \frac{1}{\sqrt{3}} \right)^2 \right] \right) \\ & + \exp \left(-8 \left[\left(x + \frac{1}{\sqrt{3}} \right)^2 + \left(y - \frac{1}{\sqrt{3}} \right)^2 + \left(z - \frac{1}{\sqrt{3}} \right)^2 \right] \right) \\ & + \exp \left(-16 \left[\left(x + \frac{1}{\sqrt{3}} \right)^2 + \left(y + \frac{1}{\sqrt{3}} \right)^2 + \left(z - \frac{1}{\sqrt{3}} \right)^2 \right] \right) \\ & + \exp \left(-10 \left[\left(x - \frac{1}{\sqrt{3}} \right)^2 + \left(y + \frac{1}{\sqrt{3}} \right)^2 + \left(z + \frac{1}{\sqrt{3}} \right)^2 \right] \right) \\ & - \exp \left(-20 (z - 1)^2 \right) - \exp \left(-15 (z + 1)^2 \right). \end{aligned}$$

Figure 4.10 illustrates this surface curl-free field. We see in Figure 4.11 that the RBF Direct method becomes numerically unstable around $\varepsilon = 1$ and that the error blows up as $\varepsilon \rightarrow 0$. We note here that the RBF-QR method not only remains numerically stable in the flat limit, but it also provides the smallest errors around $\varepsilon = 0.5$. Since

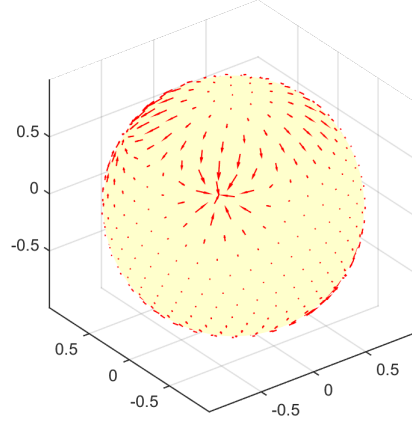


Figure 4.10: The surface curl-free vector field to be interpolated for test 2.

we showed in the previous numerical test that the vector RBF interpolant converges to a curl-free vector spherical harmonic interpolant in the flat limit, we can conclude that the RBF-QR method achieves the best approximation of this target vector field for a value of ε that is untouchable for RBF Direct.

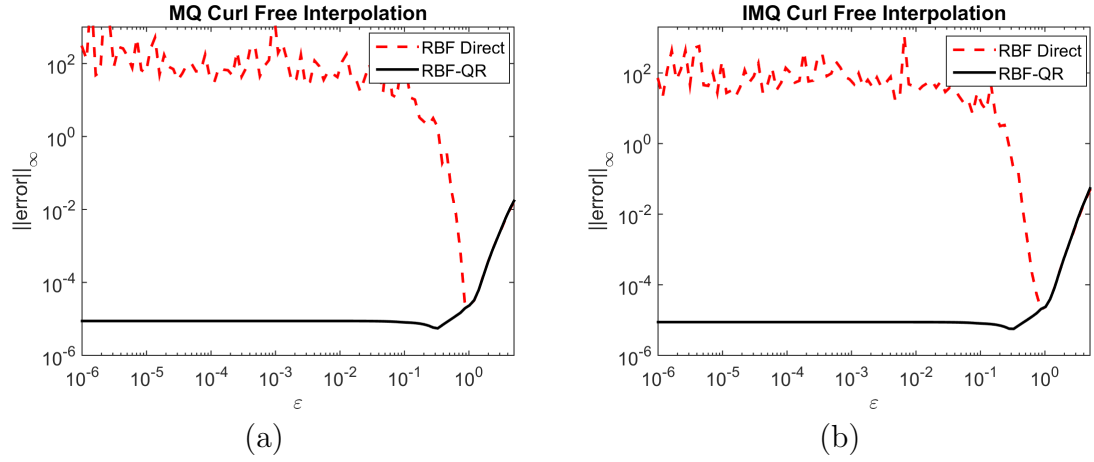


Figure 4.11: Curl-free numerical test 2: Log-log plot of the max-norm error in the approximation of the true field vs values of ε for both the RBF Direct method and the Vector RBF-QR method with the (a) MQ and (b) IMQ kernels.

4.3 Conclusions

We conclude this chapter with a brief summary of the numerical tests. First, we showed that as $\varepsilon \rightarrow 0$, the RBF-QR method converges to vector spherical harmonics in both the divergence-free and curl-free cases. Second, we showed that for smooth vector fields, the RBF-QR method with a non-zero shape parameter can provide better approximations of the target field than either the RBF Direct method or the vector spherical harmonic functions. Finally, we note that the RBF-QR method does not necessarily perform better than the RBF Direct method for rough vector fields.

CHAPTER 5

CONCLUSIONS

This work developed the first stable numerical method for calculating vector-valued RBF interpolants on the sphere in the flat limit. Modeled after the Scalar RBF-QR algorithm of Fornberg and Piret, our method bypasses ill-conditioning that is introduced into the interpolation system in the flat limit. We provide details of this development in Chapter 3, where we utilize vector-valued spherical harmonic expansions to create a new set of basis functions that both span the same space as the original RBF basis and are well-conditioned. In Chapter 4, we offer various numerical results that show the effectiveness of our algorithm in the flat limit. Additionally, these results lead to conclusions that in some cases, a vector-valued RBF interpolant with a small shape parameter can result in a better approximation of the target field than both the RBF Direct interpolant and a vector spherical harmonic interpolant. The Vector RBF-QR algorithm makes the full range of the shape parameter available for vector RBF interpolation on the sphere without concern for ill-conditioning.

Future work for this research includes modifying the Vector RBF-QR algorithm for interpolating divergence-free or curl-free vector fields without a restriction on the number of interpolation nodes, i.e. for $n \neq (\mu_0 + 1)^2 - 1$. We will also develop our algorithm for computing the Helmholtz-Hodge decomposition of a vector field based only on samples of the field at a set of n scattered points on the sphere, similar to

the method introduced by Fuselier and Wright in [13]. Additionally, we would like to develop a Vector RBF-QR algorithm for vector fields in \mathbb{R}^2 and \mathbb{R}^3 using similar techniques to those by Fornberg, Larsson, and Flyer [6].

REFERENCES

- [1] I. Amidror. Scattered data interpolation methods for electronic imaging systems: a survey. *J. Electron. Imaging.*, 11:157–176, 2002.
- [2] K. Atkinson and W. Han. *Spherical Harmonics and Approximations on the Unit Sphere: An Introduction*. Springer Berlin Heidelberg, 2012.
- [3] G. Balmino, B. Moynot, and N. Vales. Gravity field model of mars in spherical harmonics up to degree and order eighteen. *J. Geophys. Res.*, 87(B12):9735–9746, 1982.
- [4] R. E. Carlson and T. A. Foley. The parameter r^2 in multiquadric interpolation. *Comput. Math. Appl.*, 21:29–42, 1991.
- [5] B. Fornberg and N. Flyer. *A Primer on Radial Basis Functions with Applications to the Geosciences*. Society for Industrial and Applied Mathematics, Philadelphia, 2015.
- [6] B. Fornberg, E. Larsson, and N. Flyer. Stable computations with gaussian radial basis functions. *SIAM J. Sci. Comput.*, 33:869–892, 2011.
- [7] B. Fornberg and C. Piret. A stable algorithm for flat radial basis functions on a sphere. *SIAM J. Sci. Comput.*, 30:60–80, 2007.
- [8] B. Fornberg and G. B. Wright. Stable computation of multiquadric interpolants for all values of the shape parameter. *Comp. Math. Appl.*, 48:853–867, 2004.
- [9] R. Franke. *Approximation of Scattered Data for Meteorological Applications*. Birkhäuser Basel, Basel, 1990.
- [10] W. Freeden, T. Gervens, and M. Schreiner. *Constructive approximation on the sphere with applications to geomathematics*. Oxford University Press, 1998.
- [11] E. J. Fuselier. *Refined error estimates for matrix-valued radial basis functions*. PhD thesis, Texas A & M University, 2006.
- [12] E. J. Fuselier, F. J. Narcowich, J. D. Ward, and G. B. Wright. Error and stability estimates for divergence-free RBF interpolants on the sphere. *Math. Comp.*, 78:2157–2186, 2009.

- [13] E. J. Fuselier and G. B. Wright. Stability and error estimates for vector field interpolation and decomposition on the sphere with RBFs. *SIAM J. Num. Anal.*, 47:3213–3239, 2009.
- [14] N. K. Hansen and P. Coppens. Testing aspherical atom refinements on small-molecule data sets. *Acta Cryst.*, 34(6):909–921, 1978.
- [15] R. L. Hardy. Multiquadric equations of topography and other irregular surfaces. *J. Geophy. Res.*, 76:1905–1915, 1971.
- [16] R. L. Hardy. Theory and applications of the multiquadric-biharmonic method: 20 years of discovery. *Comput. Math. Appl.*, 19:163–208, 1990.
- [17] R. Horn and C. R. Johnson. *Topics in matrix analysis*. Cambridge University Press, Cambridge, 1991.
- [18] S. Hubbert and B. Baxter. Radial basis functions for the sphere. In W. Haussmann, K. Jetter, and M. Reimer, editors, *Recent Progress in Multivariate Approximation, Proc. of the 4th Intern. Conf., Witten-Bommerholz, Germany*, volume 137 of *International Series of Numerical Mathematics*, Basel, 2001. Birkhäuser.
- [19] J. P. Lewis, F. Pighin, and K. Anjyo. Scattered data interpolation and approximation for computer graphics. In *ACM SIGGRAPH ASIA 2010 Courses*, page 2. ACM, 2010.
- [20] C. A. Micchelli. Interpolation of scattered data: distance matrices and conditionally positive definite functions. *Constr. Approx.*, 2:11–12, 1986.
- [21] C. Müller. *Spherical Harmonics: Lecture Notes in Mathematics*, volume 17. Springer Berlin Heidelberg, New York, 1966.
- [22] F. J. Narcowich and J. D. Ward. Generalized hermite interpolation via matrix-valued conditionally positive definite functions. *Math. Comput.*, 63:661–687, 1994.
- [23] F. J. Narcowich, J. D. Ward, and G. B. Wright. Divergence-free RBFs on surfaces. *Fourier Anal. Appl.*, 13:643–663, 2007.
- [24] R. Nisbet, G. Miner, and J. Elder IV. *Handbook of statistical analysis and data mining applications*. Academic Press, 2009.
- [25] E. Oubel, M. Koob, C. Studholme, J. L. Dietemann, and F. Rousseau. Reconstruction of scattered data in fetal diffusion mri. *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2010*, pages 574–581, 2010.

- [26] S. Rippa. An algorithm for selecting a good value for the parameter c in radial basis function interpolation. *Adv. Comp. Math.*, 11:193–210, 1999.
- [27] G. J. Streletz, G. Gebbie, O. Kreylos, B. Hamann, L. H. Kellogg, and H. J. Spero. Interpolating sparse scattered data using flow information. *Journal of Computational Science*, 16:156–169, 2016.
- [28] P. N. Swarztrauber. The vector harmonic transform method for solving partial differential equations in spherical geometry. *Mon. Wea. Rev.*, 121:3415–3437, 1993.
- [29] H. Wendland. *Scattered Data Approximation*. Cambridge University Press, Cambridge, 2004.
- [30] J. Wojciech. *Efficient Monte Carlo Methods for Light Transport in Scattering Media*. PhD thesis, UC San Diego, 2008.
- [31] G. B. Wright. *Radial Basis Function Interpolation: Numerical and Analytical Developments*. PhD thesis, University of Colorado, Boulder, 2003.
- [32] G. B. Wright. SpherePts. <https://github.com/gradywright/spherepts/>, 2016.
- [33] G. B. Wright and B. Fornberg. Stable computation with flat radial basis functions using vector-valued rational approximations. *J. Comput. Phys.*, 331:137–156, 2017.

APPENDIX A

PROOF OF LEMMA

Horn and Johnson's book, *Topics in Matrix Analysis* [17], provides a Lemma that we utilize in both Chapters 2 and 3 when describing the fourth step of the Scalar RBF-QR and Vector RBF-QR algorithms, respectively. The proof is listed as an exercise in the book, so we provide the relevant portion here for completeness. Before we begin the proof, we must first formally define a Hadamard product, which is informally described as entry-wise multiplication.

Definition A.0.1. (Hadamard Product [17, p. 298]) The Hadamard product of m -by- n real-valued matrices A and B is defined by $A \circ B \equiv [a_{ij}b_{ij}]$, where a_{ij} and b_{ij} are the ij^{th} entries of A and B , respectively.

We now provide the part of the Lemma that is used in this thesis and continue with the proof, which uses properties of diagonal matrices and commutativity.

Lemma A.0.1. ([17, p. 304]) If A and B are m -by- n real-valued matrices, and if D and E are real-valued diagonal matrices of size m -by- m and n -by- n , respectively, then

$$D(A \circ B)E = A \circ (DBE).$$

Proof. We consider the ij^{th} entry of $D(A \circ B)E$ and show that it is equivalent to the ij^{th} entry of $A \circ (DBE)$. First we have from the definition of matrix multiplication

$$[D(A \circ B)E]_{ij} = \sum_{k=1}^m \sum_{l=1}^n [D]_{ik} [A \circ B]_{kl} [E]_{lj}.$$

Then by the definition of the Hadamard product,

$$\sum_{k=1}^m \sum_{l=1}^n [D]_{ik} [A \circ B]_{kl} [E]_{lj} = \sum_{k=1}^m \sum_{l=1}^n [D]_{ik} [A]_{kl} [B]_{kl} [E]_{lj}.$$

Since D and E are diagonal, we know that all non-diagonal entries are 0. This gives

$$\sum_{k=1}^m \sum_{l=1}^n [D]_{ik} [A]_{kl} [B]_{kl} [E]_{lj} = [D]_{ii} [A]_{ij} [B]_{ij} [E]_{jj}.$$

We can then commute these values and again use the properties of diagonal matrices to rewrite the expression as

$$[D]_{ii} [A]_{ij} [B]_{ij} [E]_{jj} = [A]_{ij} [DBE]_{ij}.$$

By the definition of the Hadamard product, we have the result

$$[A]_{ij} [DBE]_{ij} = [A \circ (DBE)]_{ij}.$$

Since each element of $D(A \circ B)E$ is equal to each element of $A \circ (DBE)$, we have equality of the two matrices. \square

We conclude with a note explicitly showing how this Lemma applies in the thesis.

In (2.10), we state

$$E_1^{-1} R_1^{-1} R_2 E_2 = (R_1^{-1} R_2) \circ (E_1^{-1} J_{n,m-n} E_2).$$

Notice that J is the identity matrix in Hadamard multiplication. Thus, if we let

$A = (R_1^{-1} R_2)$, $B = J_{n,m-n}$, $D = E_1^{-1}$, and $E = E_2$, we see that

$$\begin{aligned} E_1^{-1} R_1^{-1} R_2 E_2 &= E_1^{-1} (R_1^{-1} R_2 \circ J_{n,m-n}) E_2 \\ &= D(A \circ B)E \\ &= A \circ (DBE) \\ &= (R_1^{-1} R_2) \circ (E_1^{-1} J_{n,m-n} E_2). \end{aligned}$$